A STUDY OF NEURAL NETWORK APPLICATIONS TO ALUMINIUM MANUFACTURING

COMALCO
BELL BAY

by

Frederick Frank Frost

BE(Hons), MEngSc

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy.

School of Science and Engineering, University of Tasmania.

December 2000

To my parents, Wilfred and Betty, who taught me patience and endurance, and my wife, Terri-Ann, my inspiration and confidant.

Capturing the Power of the Human Mind

Ned Shaw

- Chemical Engineering, vol. 106, no. 4, April 1999

Declaration

This thesis contains no material which has been previously accepted for a degree or diploma by the university or any other institution, except by way of background information and has been duly acknowledged in the thesis, and to the best of the author's knowledge and belief no material has been previously published or written by any other person except where due acknowledgment is made in the text of the thesis.

Signed: .....

Dated this fifteenth day of December 2000

Authority of Access

This thesis is not to be made available for loan or copying for one year following the date this statement was signed. Following that time the thesis may be made available for loan and limited copying in accordance with the *Copyright Act 1968.*

Signed: .....

Dated this fifteenth day of December 2000

Abstract

Process behaviour in the aluminium smelting industry is typically highly dynamic and unstable and involves non-linear, highly dimensional relationships among process parameters. Further, with the presence of noise associated with most of the measured parameters of the aluminium production technique, process modelling in the aluminium industry is often a complex task. However, the advancement of both knowledge and technique has resulted in significant changes to industrial processing techniques and process control methodologies. One such advancement is the development of artificial neural networks, which are a well-suited computational paradigm for use in monitoring and controlling complex dynamic processes. Neural networks offer a powerful mathematical technique for modelling, control and optimisation of dynamic processes that are developed using process data, without the need for *a priori* knowledge or understanding the associated scientific principles and underlying relationships among process parameters. Generally, when a neural network is initially trained for a particular task, some of the features of the training data will have no significant effect on the networks decision, while other features will be critical. In addition, there exist many networks for a particular task that may perform similarly, however, they may use different features of the training data to make their decision. This work presents an evaluation and empirical performance comparison of various neural networks in an important and actual application domain. Such studies are valuable to understanding the strengths and weaknesses of various problem solving models as well as the characteristics of various application domains. As neural networks are an advanced control technique that are often used as an opportunity to maximise corporate revenue, it becomes necessary to develop a set of selection criteria for selecting a particular neural network that produces optimum performance when applied to a specific application. Neural network selection can be completed based on economic considerations, such as cost associated with neural network accuracy, cost associated with measuring process parameters used as input variables in the model and cost associated with neural network computation time. In this work, evaluation of neural networks for three industrial applications, involving process modelling of reduction cells for aluminium production at Comalco Aluminium (Bell Bay) Limited, or CABBL, is completed. The performance of six distinct models of the neural network paradigm is assessed using specific assessment criteria. The decision of which neural network model is most suitable for a specific application is complex, requiring quantitative decision logic, particularly as the assessment criteria are not fundamentally of equal significance. It is shown that optimisation techniques are necessary to select an optimum neural network model for a specific application. While it is noted that available operations research techniques are capable of neural network optimisation and selection, such optimisation techniques are inappropriate for application in this instance. This work reports a systematic technique that optimises a neural network efficiently on command using precise mathematical models. It is shown in this work that the influence of each input parameter on prediction error is analysed to determine an optimum neural network model for each studied application. Moreover, while a feasible solution for the neural network model is identified in the first instance, an optimal solution is subsequently obtained and implemented to achieve maximum economic benefit. It is noted that the developed optimisation strategy is a unique and novel methodology for neural network optimisation and selection and has been carefully developed to facilitate its ease of application in industry.

Acknowledgments

This work is the result of an ongoing collaborative relationship between the School of Science and Engineering at the University of Tasmania and Comalco Aluminium (Bell Bay) Limited. Dr Vishy Karri, Senior Lecturer in Manufacturing Engineering at the University of Tasmania, has been instrumental in developing industry links, leading to suitable industry based postgraduate research projects and mechanisms to complete the work efficiently and effectively. Moreover, Vishy receives special recognition here for his technical and moral support, guidance and recommendations provided throughout the duration of the research project. My motivation for research was significantly influenced by the genuine enthusiasm for research and advancement of knowledge exhibited by Vishy. In addition, General Manager, Comalco Aluminium (Bell Bay) Limited, David Sadler, and more recently, Duncan Hedditch, both deserve special recognition for their collaboration and support during the tenure of the research project. While David contributed resources and financial support for the project in the first instance, Duncan continued this support and for doing so is greatly appreciated. The support given by industry has significantly contributed to the successful completion of this research project. Further, specific technical support relating to the process considered has also been provided by Kyle Gimpl, Manager - Potrooms, Comalco Aluminium (Bell Bay) Limited, and Mark Taylor, General Manager - Technical, Comalco, who both deserve special acknowledgment for their contribution. Typically with industry based research it is necessary to validate research results through application. The work completed in this instance was no exception, with significant contribution from technical staff at the Bell Bay smelter. In particular, Kaye Saunders and Kaija Russack provided valuable expertise throughout various stages of the project, while Rob Clifford, Greg Picot and Robbie Matthews each contributed specific knowledge and recommendations to facilitate and enhance the research work. Further, the plant statistician, Leisa Kammholz, deserves special recognition for her recommendations and guidance regarding appropriate statistical techniques for analysis of results. While available literature provides substantial information on the modelling paradigm studied as a part of this work it is always useful to have resources available to answer specific questions. In this instance, Warren Sarle, through personal e-mail and his on-line FAQ series on neural networks, has provided extremely useful information on several occasions during the research work. Likewise, Professor David Lowe, developer of the radial basis function neural network, supplied valuable information via personal e-mail and forwarded literature when requested and for doing so is greatly appreciated.

Accompanying Software

A compact disk (CD) containing specific computer programs, formatted descriptive program code and relevant published refereed international journal and conference papers is provided with this thesis. In particular, the disk contains executable code for seven (7) neural network programs, for which details are provided in the main body of the thesis. Each of the neural network programs are written in Pascal program code and are executable using Turbo Pascal, version 6.0, or later. In addition, formatted descriptive source code for each of the neural networks is included on the disk, readable using Microsoft® Word, version 97 SR-2, or later. A user guide for the neural network programs is provided in Appendix A. A neural network pre-processing and optimisation program is also included on the disk. The program is written in Microsoft® Visual Basic™ program code and is executable using Microsoft® Excel, version 97 SR-2, or later. Formatted descriptive source code for the program is also included on the disk, readable using Microsoft® Word, version 97 SR-2, or later. A user guide for the neural network pre-processing and optimisation program is provided in Appendix D. Further, the disk contains relevant published refereed conference and journal papers, readable using Microsoft® Word, version 97 SR-2, or later. Refer to Appendix H for full publication details of the accompanying conference and journal papers.

Contents

CHAPTER ONE

Introduction

The advancement of both knowledge and technique has resulted in significant changes to industrial processing techniques and process control methodologies, generally resulting in improved process efficiency. Continual improvement is an ever present theme in industry, suggesting that processes are continually being optimised, processing costs reduced, production rate and capacity increased, while maintaining or improving the quality of the final product and without compromising workplace health and safety standards. In order to remain competitive in today's global economy, industries must strive to perform at an extremely high level. Professor Alan Gilbert [1] made a noteworthy statement that has substantial relevance to industry, "only in sport do Australians seriously accept that if you are not as strong, as fast, as nimble, as determined and as skilled as the best in the world, then you lose". This concept is directly applicable to industry, in that companies must maximise success by enhancing their competitiveness. Industries must place themselves in an economically sound position, continually striving to operate under optimum performance conditions, in order to remain competitive and exist in the international domain. To achieve this, industrial organisations must examine all elements of product flow and develop improved techniques and procedures to carry out these events efficiently and effectively. Generally, if a process, which can be simply defined as an advance to some particular objective through an orderly sequence of operations, is to be worthwhile it must be carried out efficiently. In order to maintain efficiency, control of the process is necessary throughout the complete cycle of operations. In most circumstances, this process control cannot be maintained by human endeavour alone. It is necessary at least to have some instrumental assistance to measure the process variables as an indicator of process behaviour. Further, computers and automated systems are becoming increasingly popular as industry technology advances. The use of integrated computer control and the rapid growth in technology has allowed industry to respond faster to process changes, meet

challenging customer demands and remain viable and competitive in the changing international market. However, an important feature common to all processes is that a process is very rarely in a stable static equilibrium for more than a very short period of time. Rather, a process is a dynamic entity subject to continual changes and disturbances, which influence the process to deviate from some desired state of equilibrium. Consequently, corrective action is required to return the process to the desired state. While some disturbances exhibit cyclic behaviour, which influence the process to behave in a periodic manner, most disturbances arise externally to the process and are of transient behaviour, fading away with no likelihood of recurrence. This makes it extremely difficult to predict process response to a given situation and consequently, the appropriate corrective action is not generally realised until after the disturbance event has occurred. Nevertheless, the main objective of process control is to maintain process behaviour within some bounded range. This bounded range is usually in the form of a control chart with upper and lower control limits for which endeavours are made to maintain the process within. However, random process variation and the occurrence of large unpredicted disturbances affect the process occasionally to cause deviation from the control range. Hence, it is the focus of developments in the field of process control to develop techniques to monitor accurately process variation and better prepare for process deviations. It is useful to note that the ultimate goal of any process control effort is to minimise variation and deviation of the process from some desired value or range. In developing a control system for a particular process it is necessary to determine those variables which are to be controlled in order to maintain some desired level of efficiency. The output then of the control system is the value of the specific variable to be controlled, which is not necessarily the same as the output of the process. While the process output is generally a physical quantity, the output of the control system need not necessarily be a product of the process.

A mandatory requirement for the design procedure of any control system is a description of the process to be controlled, generally in the form of a mathematical model. While modelling from data is an integrated part of process development, model building based upon scientific principles of the process is an extremely challenging task in such an environment where a quick solution is required and *a priori* knowledge of the process to be modelled is limited. Due to the general

complexity of process systems, a considerable amount of time must be devoted to mechanistic modelling and moreover, simplifying assumptions have to be made in many circumstances to enable some type of solution to the modelling problem. This generally leads to costly solutions that are often inaccurate due to the assumptions made. Therefore, phenomenological models, which represent the external behaviour of a process based on a system of algebraic equations without any consideration of the associated scientific principles, are often necessary. Phenomenological models, or empirical models, are usually developed relatively quickly using historical process data and process observations. Such models are useful as they allow an evaluation and analysis of the process behaviour through computer simulation. In addition, they provide a means by which processes can be controlled and optimised with an understanding only of the overall external behaviour of the process, without any knowledge of the underlying scientific basis. Hence, the application of a technique that facilitates rapid and cheap development and is capable of learning and expressing the non-linearities and complexities of a particular process becomes a desirable objective. Artificial neural networks possess this ability and consequently are well suited to many applications of process control. Neural networks have the ability to be trained to accurately simulate process behaviour based on observations of the system over time and in response to external actions. Neural networks develop their own functional relationship to model the process, which can then be used to determine how the process will evolve from some initial state or respond to an applied external action. During the learning process, the neural network is continually modifying its computational model, developing an accurate description of the process, converging to an optimum performing model, yielding the best model of the dynamic system. Once an accurate model of the process is achieved, predictive monitoring is possible, obtaining the value of a particular hard-to-measure parameter, or parameters, based on the values of known related parameters, that are generally easier to measure.

While regression is the traditional approach to empirical modelling, neural networks posses certain desirable features that enable them to exceed the limitations of traditional information processing techniques. The growing interest and applications of neural networks is greatly due to the desirable properties that these devices offer. One such property that is fundamental in the architecture of neural networks is

parallelism. This particular property is desirable as it gives rise to a significantly increased calculation speed compared with traditional information processing techniques. Another of these properties is the capacity for adaptation that neural networks possess. This particular property enables neural networks to take account of new constraints and new data as they arise. The implications of this are that neural networks are able to adapt to changes in the problems they are solving. Further, neural networks operate using distributed memory, a particularly desirable feature of information processing. The loss of an individual component of data does not cause the loss of an entire stored data item. This differs to traditional computers in which the loss of memory causes data items to be permanently lost. In neural networks the loss of a memory unit only marginally changes the ability of the network to process data accurately. Distributed memory is a desirable feature of neural networks as it is possible to obtain accurate data from the network even when noisy data is presented. This also leads to a further important feature of neural networks, capacity for generalisation. This particular feature is of significance as it allows neural networks to simulate the behaviour of a particular process based only on a small number of representative examples. Once trained using these examples neural networks are able to generalise for a given situation not presented in the training data. Further, neural networks are considered for many applications due to the relative ease with which they can be implemented. Computer simulation programs that direct general purpose digital computers to perform the actions and characteristics of a neural network require only a short development time and are relatively simple to implement. For practical applications there exist three distinct methods for implementation of neural networks; software simulation programs, special purpose digital network simulators and true electronic or optical models. The sequence in which these options are presented is significant as it highlights the relative ease with which neural networks can be implemented. For instance, a software simulation program is generally the cheapest and fastest implementation method while the third option involves the lengthy process of implementing the networks physical components using associated hardware.

While there are many commercial neural network software simulation packages available it is often necessary and useful to write neural network program code oneself. Generally, commercial neural network packages do not allow access to the

source code, making it extremely difficult to make modifications to a network to better suit a particular application. Further, neural network implementation requires the code to be compatible with the database from which input values for the network are stored. Hence, this often requires translating the neural simulator into the particular code used within the industry. Hence, it is critical that the neural network source code be accessible. For these reasons, the author has written a series of neural network programs that are used as an integral part of this research project. Moreover, all neural network programs used in this research have been written using Pascal programming language. Neural network programming involves straightforward numerical computation, hence, the properties of Pascal, including clarity and readability, make this particular programming language suitable for the task. Pascal contains all the necessary mathematical functions required for neural network computation and the functions and procedures written using this code are relatively easy to understand.

As part of an investigation into the ability of neural networks to accurately model process behaviour, some practical applications in a dynamic industrial environment are considered. The particular industry considered in this work is Comalco Aluminium (Bell Bay) Limited, or CABBL, situated at Bell Bay in northern Tasmania, Australia. CABBL are majority owned by Rio Tinto, approximately 72.0%, and the remainder by financial institutions and individuals, principally in Australia and New Zealand. Comalco, which is the name given to the parent company incorporating CABBL, Boyne Smelter Limited (BSL) in Queensland, Australia, and New Zealand Aluminium Smelter (NZAS) in Invercargill, New Zealand, was established in 1955 following the discovery of massive bauxite deposits in Weipa, Australia. Comalco produce approximately 26.0% of the primary aluminium in Australia and 100.0% of the primary aluminium in New Zealand. Comalco are a major Australian exporter with sales totalling more than $1.0 billion annually. With market capitalisation of $3.5 billion Comalco is one of Australia's leading publicly listed companies. In regard to strategic direction, research and the use of leading edge technology are important to achieving Comalco's goals. Comalco is committed to applying new technologies to improve the efficiency of its production processes. The particular smelter being considered in this instance, CABBL, commenced production in 1955 and is the oldest of the three aluminium

smelters operated by Comalco throughout the Australasian region. Aluminium production output from the plant has increased from an initial capacity of 12,000 tonnes in 1955 to a current capacity of 160,000 tonnes. An aerial view of the aluminium smelting plant at Bell Bay is shown in Figure 1.1.1, indicating the physical size of the smelter and some of the 2,600 hectares of company owned land surrounding the plant. Comalco has a strict continuing policy towards environmental sustainability. It is committed to the highest possible environmental performance that can be achieved with the existing technology.

Fig. 1.1.1. Aerial Photograph of Comalco Aluminium (Bell Bay) Limited Highlighting the Smelter and Immediate Surroundings

While a brief overview of the company is provided here, the specific details and particularities of the aluminium production process at CABBL are complex and require considerable discussion and therefore are given in the following literature survey. It is also useful to note here that all monetary values given in this dissertation are expressed in Australian dollars (AUD), unless otherwise stated.

CHAPTER TWO

Literature Survey

2.1 INTRODUCTION TO ALUMINIUM

Aluminium is the third most abundant element in Earth's crust, following iron and silicon, and is present in the crystal structure of many rock forming elements. Due to its chemical reactivity it is found in combination with other elements, commonly in an oxidised form known as alumina, Al_2O_3. Aluminium has a range of characteristics that make it an extremely useful and desirable material. One of the most outstanding characteristics of aluminium is its light weight, with a density of 2,700kg/m^3, it weighs only approximately one-third as much as the same volume of steel, copper or brass [2]. Other notable features of aluminium are its excellent electrical and thermal conductivity, high corrosion resistance and its ability to form high strength when alloyed with other elements, such as silicon and magnesium. Aluminium and many of its alloys can be worked readily into any form needed, be cast by most foundry processes and accept a wide variety of attractive, durable and functional surface finishes [3].

2.1.1 Applications and Importance of Aluminium

The unique combinations of properties provided by aluminium and its alloys make aluminium one of the most versatile, economical and attractive metallic materials for a broad range of uses from soft, highly ductile wrapping foil to the most demanding engineering applications [3]. Light weight is the dominating factor for using aluminium for transportation equipment and movable parts, while an important factor contributing to the use of aluminium for exposed structures is the low cost of keeping the metal presentable in appearance and structurally sound. In addition, the fact that the strength of aluminium can be substantially increased through alloying and controlled heat treatment also leads to aluminium being an extremely competitive metal for applications where steel was predominantly used. Further, aluminium is easily machined and fabricated compared to other materials due to its high

malleability and can be cast by all foundry processes. Aluminium has high electrical and thermal conductivities, is highly reflective to radiant energy-visible light, radiant heat and electromagnetic waves and is nonferromagnetic [3]. This range of properties makes aluminium particularly useful in the electrical and electronic industries for a variety of applications. Aluminium and its alloys find use in many domestic, commercial and industrial applications, some of which are shown in Figure 2.1.1.

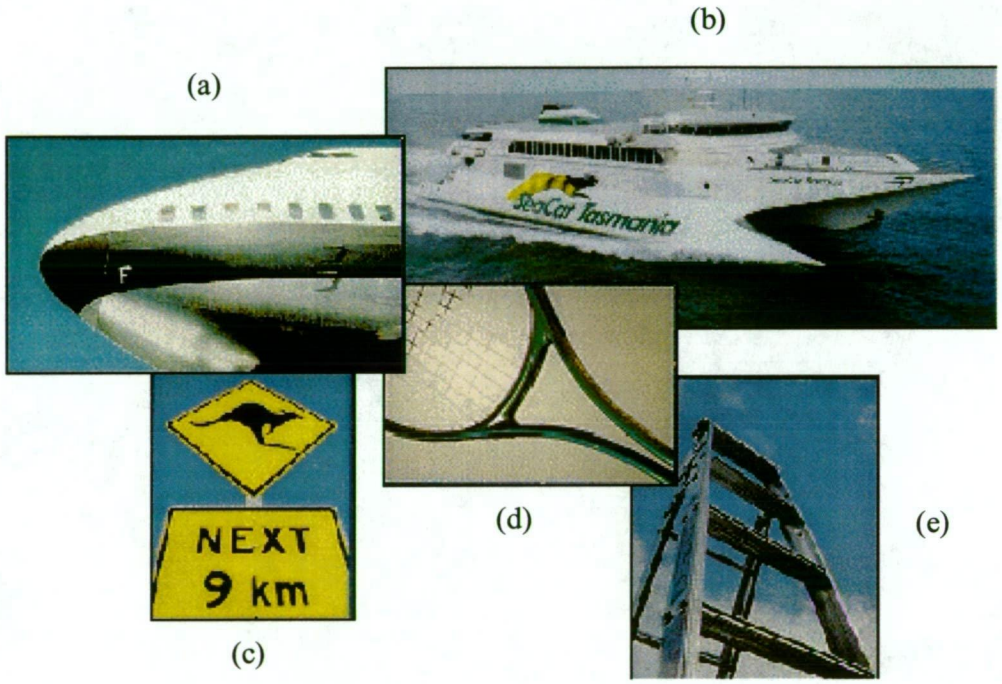


Fig. 2.1.1. Some Typical Applications of Aluminium. (a) Aeroplane Structures and Components, (b) Sea Ferry Structures and Components, (c) Road Signs, (d) Tennis Racquets, and (e) Ladders.

Among the commercial metals, aluminium is second only to iron in production and consumption on a weight as well as a volume basis [2]. Mill products constitute the major share of total aluminium product shipments, followed by castings and ingot other than for castings [2]. The Australian aluminium industry is an integrated industry with bauxite mining, smelting, casting and semi-fabrication facilities. The Australian primary aluminium industry has grown to a capacity of almost 1.8 million tonnes per year and now ranks as the world's fourth largest producer after the US (3.8 million tonnes), USSR (3.2 million tonnes) and Canada (2.1 million tonnes) [4]. The Australian aluminium industry directly employs approximately 15,000 people, mainly in regional areas and in 1996 the industry generated over $5.0 billion in

export earnings and was, as a whole, Australia's second largest export industry, behind coal [5]. Currently the major export markets for Australian primary aluminium are in the Pacific Rim, mainly Japan, to which Australia exports about two thirds of its aluminium production [6]. Consumption of aluminium throughout Australia is increasing and is now over 350,000 tonnes per year [5].

2.1.2 The Interesting History of Aluminium

Although 'alum', the element from which aluminium takes its name, had been in use for centuries, it was not until 1722 that Friedrich Hoffman concluded from experiments that the base of alum was alumina. Convinced that alumina had a metallic base, in 1807 Sir Humphrey Davy attempted unsuccessfully to reduce it by heating it with potash and an electrolysing mixture [2]. Although unsuccessful, Davy initially named the material 'alumium' and then later changed it to 'aluminum' to coincide with the oxide name alumina. Consequently, the name 'aluminum' is used in the United States of America today while 'aluminium' is accepted as the name of the metal in most other countries. The next major development in the history of aluminium was in 1825 when Danish scientist H. C. Oersted announced to the Royal Danish Academy of Sciences that he had obtained the "metal of clay". Oersted achieved this by heating potassium amalgam with aluminium chloride, and distilling the mercury from the resultant aluminium amalgam, he produced a small lump of metal with the colour and lustre of tin. Developments in chemical production of aluminium continued through to 1854 when H. Sainte-Claire Deville produced aluminium by mixing sodium and aluminium chloride. The size of the pieces produced by Sainte-Claire Deville were described at the time as mammoth pieces, although only approximately the size of marbles, as they were around one thousand times larger than the previously available pieces of metal [2]. This work triggered great experimental activity in the field and led to the production of bar and ingot form of the metal in 1855. Further developments focussed on reducing the cost of producing the metal through to 1886 when Eugene and Alfred Cowles patented a process of producing aluminium alloys by electrothermal reduction of a mixture of alumina, carbon and a heavy metal, such as copper [2]. In 1886 Cowles was using the process in a plant in New York to produce a copper-aluminium alloy containing approximately 10.0 to 20.0% aluminium.


C. M. Hall

However, the process for the production of aluminium was revolutionised in early 1886 when American Charles Hall and Frenchman Paul Heroult simultaneously but independently discovered an electrolytic process that allowed an efficient separation of the metal, known today as the Hall-Heroult process [7]. In February 1886, thirteen days after Hall had discovered molten cryolite could be used to dissolve alumina, he built a prototype carbon cell in which he passed a direct current of electricity through a molten bath of dissolved alumina and produced aluminium. Without any knowledge of Halls' discovery Heroult had developed an identical process, which he patented in April 1886. Although Heroult had the earlier patent by eleven weeks, Hall was able to confirm that his discovery preceded Heroult's patent by eight weeks, through postmarked letters written to his brother detailing the discovery. While Hall's priority was accepted by the American patent examiner it was not accepted by the French, and litigation ensued. Eventually an agreement was reached whereby Hall had the American rights and Heroult the European rights [8]. The advantage of the Hall-Heroult process was that cryolite offered a reasonably low melting point, a low operating voltage and specific gravity sufficiently low for the reduced aluminium to sink to the bottom of the cell, through the electrolyte [2]. The discovery of the Hall-Heroult process triggered commercial production of metallic aluminium in 1889. This involved the electrolysis of a solution of aluminium oxide dissolved in molten cryolite at approximately 975.0°C [9]. Despite years of intense research since the discovery of the Hall-Heroult process, no economic alternative has ever been found to replace it [8].


P. Heroult

2.1.3 Production of Aluminium from 'Then' Bauxite

Aluminium ore, commonly named bauxite, is refined into aluminium oxide trihydrate, commonly named alumina, and then electrolytically reduced into metallic aluminium. Hence, production of aluminium is essentially a three-stage process, involving mining of bauxite, refining of bauxite to alumina and smelting of alumina

to produce primary aluminium metal [4]. The physical form of the material at each stage of the process is illustrated in Figure 2.1.2, from which it can be seen the raw material undergoes significant changes in form during the conversion process from bauxite to aluminium. It is shown that the initial coarse grain bauxite, which is deep red in colour, is converted to a fine white powder, alumina, and finally to the shiny solid aluminium ingot. It is useful to provide a brief note on bauxite, alumina and primary and secondary aluminium in the following section prior to detailing the particularities of the production process for aluminium.

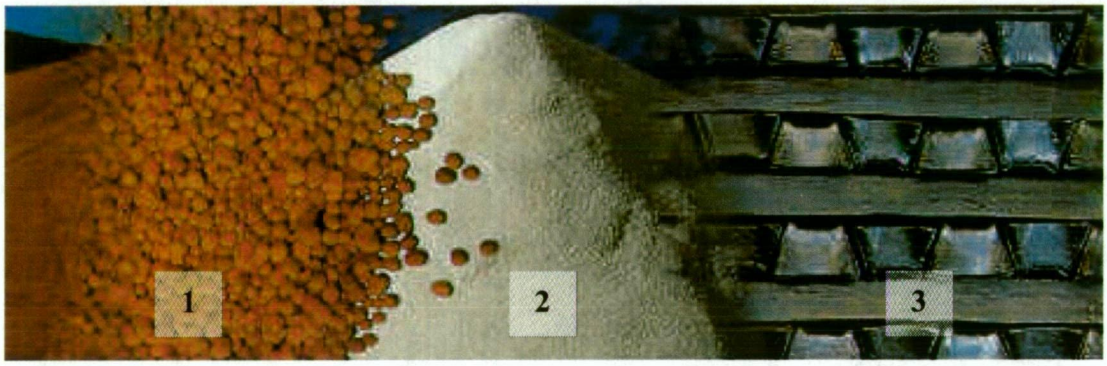


Fig. 2.1.2. Illustration of Physical Solid Form of (1) Bauxite, (2) Alumina and (3) Aluminium

Bauxite - The principal current source of aluminium is the mineral bauxite, from which aluminium oxide, or alumina, is extracted and prepared for the smelter by crushing, grinding, chemical processing and calcination [3]. Although aluminium ores are widely distributed in Earth's crust in non-bauxite ores such as kaolin clays, only bauxite has proved economical as a source of ore from which aluminium can be smelted [10]. Hydrated forms of aluminium oxides with various impurities such as iron oxides, titania and silica are contained within bauxite ores. As bauxite usually occurs in shallow near surface deposits, mining of the mineral is relatively simple. Topsoil and vegetation are generally replaced once bauxite mining is complete. It is interesting to note that approximately two to three tonnes of bauxite are required to make one tonne of alumina, depending on the grade of the bauxite.

Alumina - Due to impurities present in bauxite ores, a chemical process, known as the Bayer process [11], is used to extract alumina, as shown in Figure 2.1.3.

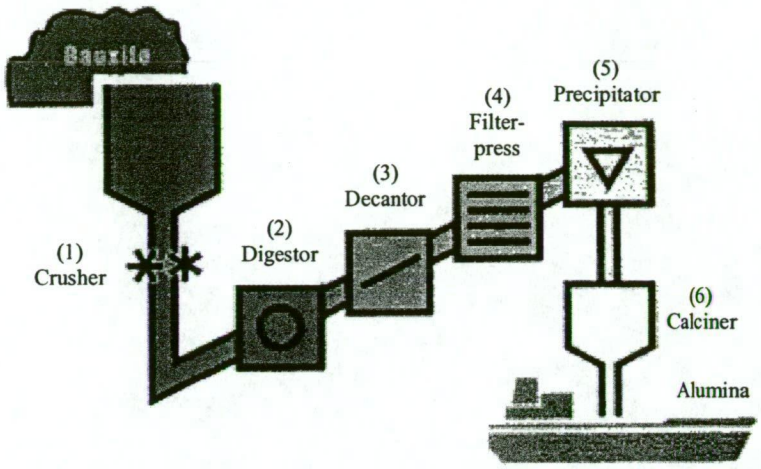
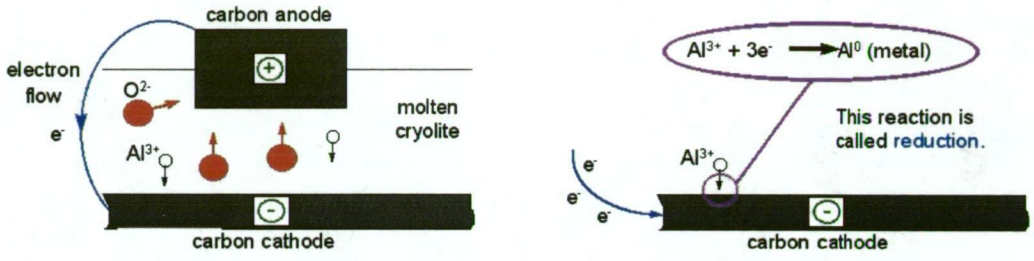


Fig. 2.1.3. Process Flow Chart Highlighting Fundamentals of Bayer Process for Conversion of Bauxite to Alumina [12]

Referring to the Bayer process flow chart, it is shown that bauxite is ground (1) and mixed with recycled spent liquor and the mixture is digested (2) to dissolve the alumina minerals. The output slurry from the digester consisting of pregnant liquor and insoluble residue, or red mud, is cooled (3) prior to the red mud being separated by settling and filtration (4). The red mud is washed free of caustic soda and discarded. The pregnant liquor is further cooled and passed to a precipitation process (5) where aluminium hydrate is deposited. The hydrate from this operation is classified into product and seed fractions. The seed fraction is recycled to the precipitation process. The product fraction is washed free of caustic liquor and calcined (6) to alumina. Because the mechanical properties of aluminium are strongly influenced by small amounts of alloying agents, it is important that the primary aluminium metal is of a high quality, hence, the smelting of pure aluminium requires high purity alumina to be used [13]. It is interesting to note that approximately two tonnes of alumina are required to make one tonne of aluminium.

Aluminium - Direct-current electrolysis of aluminium oxide dissolved in a molten sodium fluoride-aluminium fluoride bath at temperatures of 940.0 to 980.0°C is used to produce primary aluminium. A positively charged carbon anode and negatively charged carbon cathode, as shown in Figure 2.1.4(a), are used to generate a flow of

electrical current that results in molten aluminium being deposited on the cathode surface, as illustrated in Figure 2.1.4(b).



(a) (b)

Fig. 2.1.4. Illustration of (a) Direct-Current Electrolysis Process, and (b) Aluminium Deposited on Carbon Cathode as a Result of Electrolysis

It has been noted in a previous section that the technique for the production of primary aluminium is known as the Hall-Heroult process. Due to the complex particularities and characteristics of aluminium production using this technique the following section is dedicated to a detailed discussion of the Hall-Heroult process. However, it is interesting to note here that while at the current primary aluminium production level known bauxite reserves will last for hundreds of years, products made from aluminium can be recycled repeatedly to produce new products. The increasing use of recycled metal saves both energy and mineral resources needed for primary production. Secondary aluminium is the name given to aluminium recovered from scrap and is an important contributor to the total metal supply. Scrap may be recovered from either plants making end products or metal that has been previously used by consumers and is classified as either new or old scrap, respectively. Recycling of scrap metal is important from an economic point of view. The energy required to remelt secondary aluminium in preparation for reuse is only five percent of that required to produce new aluminium [6]. Hence, many aluminium industries have developed their own processing units to produce secondary aluminium.

2.1.4 Hall-Heroult Process for Aluminium Production

It has been noted that the Hall-Heroult process is an electrolysis process involving the conversion of bauxite ore to aluminium metal. Electrolysis is an electrochemical process by which electrical energy is used to promote chemical reactions that occur at

electrodes. The anode involves the oxidation process where species lose electrons, while gaining of electrons occurs at the cathode [14]. Electrolysis often involves metals that are capable of being reduced at the cathode to metallic atoms, these atoms can then be deposited on whatever is serving as the cathode surface [14]. Since its discovery, the Hall-Heroult process has remained virtually unchanged in principal. In this process pure alumina is dissolved in a bath of molten cryolite in large electrolytic furnaces, called reduction cells, or 'pots', as shown in Figure 2.1.5.

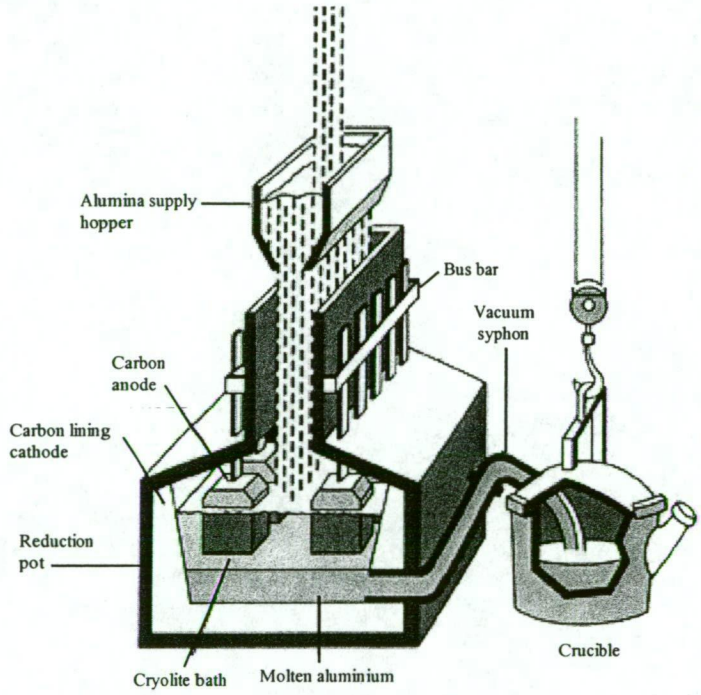


Fig. 2.1.5. Basic Components of the Reduction Cell Highlighting Removal of Molten Aluminium [16]

Molten cryolite, having a high solubility for aluminium oxide, is the major component of the Hall-Heroult electrolyte [15]. By means of a carbon anode suspended in the bath, electric current is passed through the bath mixture causing metallic aluminium to be deposited on the carbon cathode at the bottom of the cell. The heat generated by passage of this electric current keeps the bath molten, so that alumina can be added as necessary to make the process continuous. At intervals aluminium is siphoned from the reduction cells. Metal removal from the reduction cell, referred to as 'tapping', is one of the routine operating procedures and is usually completed on a daily basis by syphoning the metal into a transportable vessel. For stable cell operation, it is important that the amount of metal removed balances the

production in the time interval [13]. The metal produced is of high purity, typically 99.8% aluminium or higher. The aluminium metal removed from the reduction cell is in a molten state and is generally processed further into solid aluminium ingots or billet, packaged for shipping and sent to respective customers. The efficiency with which aluminium metal can be produced contributes significantly to the economic status and existence of the producer.

The major chemical reaction occurring in the reduction cell is the conversion of alumina and carbon, by means of electricity and electrolytic solution, to aluminium metal and consequently, some carbon dioxide. This chemical process is written as:

$$2Al_2O_{3\,(solution)} + 3C_{\,(solid)} = 4Al_{\,(liquid)} + 3CO_{2\,(gas)} \qquad (2.1.1)$$

While there has been no change to the methodology of the Hall-Heroult process there has been significant changes to the design and configuration of the reduction cell, often as a result of a strive for improvements in operating efficiency. In order to understand better the Hall-Heroult process it is useful to consider the particularities of the reduction cell.

2.1.5 Reduction Cell for Aluminium Production

The reduction cell for aluminium production incorporates a refractory vessel, super structure and electrical busbar system, as shown in Figure 2.1.6.

Refractory Vessel - The refractory vessel is used to contain the process liquids. This vessel consists of a steel shell, strengthened by cradles and a deck-plate, which is lined with a range of refractory materials chosen for either electrical conductivity, thermal insulation properties, or resistance to corrosion of the cell contents. The floor of the cell, the cathode, consists of pre-baked carbon blocks prepared from anthracite or graphite, below which are placed selected refractory materials to provide thermal insulation. The sidewalls of the shell are lined with materials offering corrosion resistance to the cell contents, typically silicon carbide or carbon.

Super Structure - The super structure is positioned above the refractory vessel and serves the following functions:

i). a suspension frame for the carbon anodes, with a means to raise and lower the anodes as required for control of the process

ii). a system for feeding alumina to the cell, comprising temporary alumina storage (alumina hopper), pneumatic breakers (crust breakers) to penetrate the cell crust and an alumina discharge device (feeder)

iii). a fume containment and discharge system, including hooding which is removed temporarily to perform routine cell operations

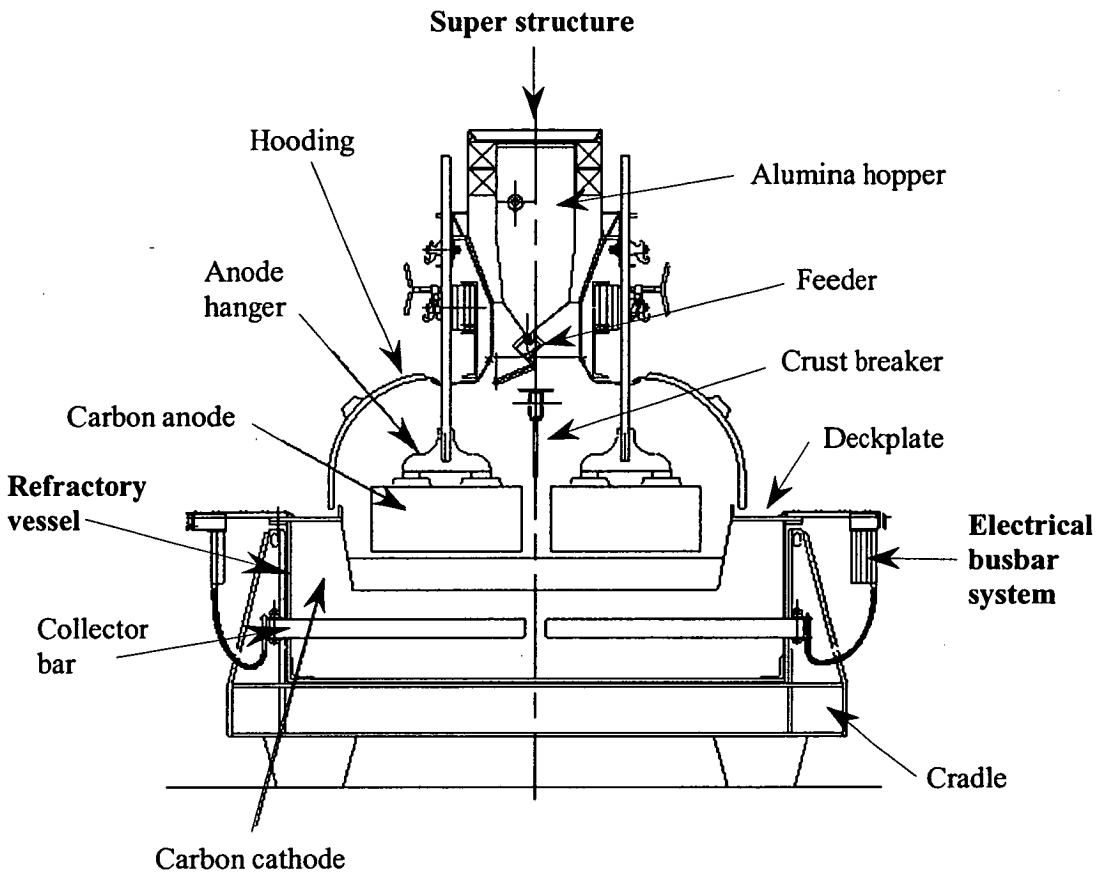


Fig. 2.1.6. Cross-Sectional End View of Hall-Heroult Cell Highlighting Refractory Vessel, Super Structure and Electrical Busbar System

Electrical Busbar System - The electrical busbar system provides for the passage of electrical current through the cell to allow the electrolysis reaction to proceed. The principal busbar elements are the anode bus, which takes current from the up-stream cell in the electrical series and delivers it to the carbon anodes and the cathode bus, which takes current from the cathode carbon blocks to the cathode ring bus and delivers it to the down-stream cell in the series. Commonly, 150 to 200 cells are

connected in series to allow the use of high voltage rectifiers. The series of connected cells, as shown in Figure 2.1.7, are referred to as a reduction line, or more commonly, a 'potline'. All cells on a potline have exactly the same electrical current flowing through them as a result of being connected in series.

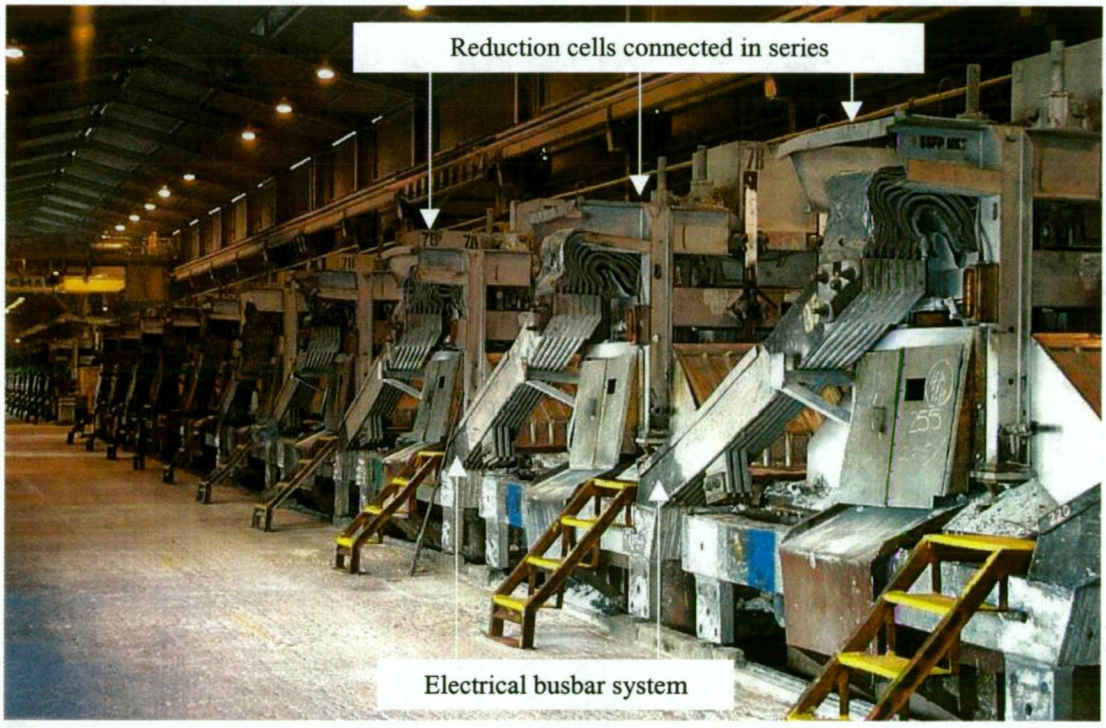


Fig. 2.1.7. Potline Section Highlighting Reduction Cell Connection in Series via Electrical Busbar System

While the main components of the reduction cell have been discussed it is necessary to note that two important phenomena that occur within the reduction cell during aluminium production are 'ledge' and 'sludge'. It is useful to provide some description of these phenomena in the following section, as they will be referred to later in the thesis.

Ledge - Aluminium reduction cells are intentionally operated at temperatures which allow some bath material to freeze onto the sidewall lining. This is achieved by controlling the input power to each cell. This layer of frozen bath material, shown in Figure 2.1.8, is called ledge and prevents contact between the sidewall lining and the highly corrosive liquid bath. It is essential to have ledge protection as none of the existing sidewall lining materials are able to withstand the corrosive and erosive

nature of the turbulent bath over a long period of time [17]. In addition, ledge acts as a temperature stabiliser. If the bath temperature rises, some ledge melts, consuming heat and reducing the severity of the temperature rise. On the other hand, if the bath temperature drops, some ledge freezes, releasing heat and reducing the severity of the temperature drop. Further, the ledge provides insulation in the cell, thereby minimising loss of heat and volatile fluorides.

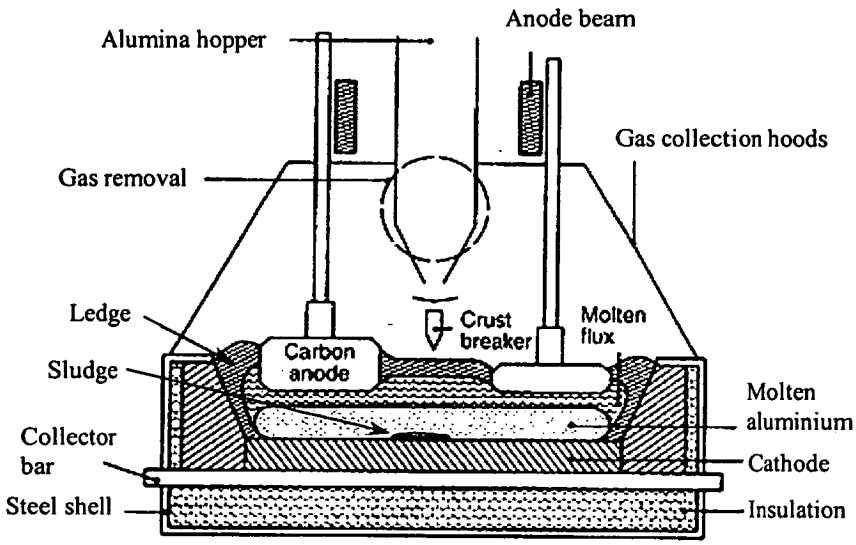


Fig. 2.1.8. Cross-Sectional End View of Hall-Heroult Cell Highlighting Ledge and Sludge [18]

Sludge - Sludge is the name given to the formation of undissolved alumina and saturated electrolyte that accumulates on the surface between the metal pad and the carbon cathode, as shown in Figure 2.1.8. The presence of sludge on the carbon cathode is undesirable for cell operations as it causes voltage instability and possibly changes the current distribution and thermal profile of the cell, due to increased insulation on the cathode [17]. The main source for sludge formation is undissolved alumina.

2.1.6 Reduction Cell Processes

The industrial reduction cell is a dynamic environment where many reactions are occurring simultaneously. The fundamental processes occurring in the reduction cell are summarised as follows [19]:

i). reduction of dissolved alumina by electrolysis, forming liquid aluminium on the cathode of the cell

ii). oxidation of the carbon anode by the oxygen released from the alumina, to form carbon dioxide gas and a smaller proportion of carbon monoxide

iii). volatilisation of electrolyte components, which are commonly scrubbed and recycled to the cell using the alumina which is fed to the process

iv). deterioration of the carbon cathode lining, predominantly by the formation of aluminium carbide, over long time periods

It is these fundamental processes that give rise to routine reduction cell operations, such as control of the alumina concentration in the electrolyte, replacement of carbon anodes due to continuous consumption, regulation of cell resistance, periodic removal of aluminium from the cell and cathode relining due to deterioration over long periods [19]. Reducing the frequency and magnitude of cell maintenance operations forms the basis of smelter process technology improvements. Continual improvements are sought for reducing energy requirements and costs per unit of aluminium metal produced. Although electricity remains the main operating cost in aluminium smelters, averaging about one quarter of the total on average, improvements in operating efficiency have reduced the per unit input of electricity in some new plants to about 13,500 kilowatt-hours per tonne of primary aluminium produced [4].

In order to produce quality primary aluminium metal there are specific materials that must be consumed as part of the Hall-Heroult process. In particular, consumable materials may leave the cell with the liquid metal, as gas or entrained particulate emissions, irreversibly enter the cathode lining or may be removed from the cell during a routine operation, such as anode changing [13]. Due to the continuous consumption of some process materials it is necessary to periodically add particular substances to the cell in order to maintain stable operation. These materials include alumina, carbon, electrolyte additives and electricity. It is useful to give a brief note on the importance of each of these important materials and highlight their role in the production of aluminium.

Alumina - As alumina is continuously consumed during the production of aluminium it must be fed to the cell at a similar rate to consumption. The alumina concentration in the electrolyte must be kept above the limit of which anode affect [13, 20] occurs (~1.5%) and below the saturation limit where sludge forms (~6.0%). Within this range there is an optimum concentration where the cell resistance is minimised [19]. While the positive effects of increased alumina concentration are lower liquidus temperature of the electrolyte, lower bath density and lower fluoride emissions, negative effects such as sludge formation and dissolution become more critical at higher alumina concentrations [17]. Hence, it is critical for stable operation to maintain a desired alumina level in the cell. Up to 60.0% of the total heat loss from a cell occurs through the top crust and anode cover. The consumption and integrity of the crust material is subject to cell operating conditions and is a major drive of the thermal imbalances which occur during operation. An effective cell cover is also necessary to limit particulate and gaseous fluoride emissions from the cell [19]. In order to make alumina additions to the reduction cell, a steel beam breaks the crust, as shown in Figure 2.1.8, and simultaneously pushes the overlaying alumina into the electrolyte. Subsequently, a predetermined amount of alumina is added to the cell and the crust is reformed.

Carbon - While it is noted that carbon is present in the reduction cell as a permanent cathode, carbon is also added to the process periodically in the form of an anode. Traditionally, carbon has been used in this capacity due to its ability to withstand the corrosive forces of the melt while giving rise to an environmentally acceptable gaseous product [21]. Anodes are consumed during metal production at a rate of around 0.39 to 0.45 kilograms of carbon per kilogram of aluminium produced, representing approximately 19.0% of the total production cost of aluminium [22]. Carbon anodes are consumed in the process by the reduction of oxygen containing anions at the anode surface to produce carbon dioxide. In addition, carbon is also lost through direct oxidation by the air above the cell [23]. Carbon anodes provide the means for electrical transport to the cell and act as the redundant in the electrochemical process.

Electrolyte Additives - The electrolyte, also commonly called 'bath', used in industrial reduction cells for the production of aluminium is composed primarily of

molten cryolite, Na_3AlF_6, which has the unique property of having a higher solubility, necessary for electrolysis, than any other fluoride compound [24]. Na_3AlF_6 melts at 1010.0 ± 2.0°C, is a good solvent for alumina and is thermodynamically very stable allowing for the electrolytic decomposition of alumina [25]. The choice of bath composition has been influenced in many cases by tradition and background of the reduction cell operators. However, the general trend has been towards bath compositions that give lower bath temperatures and thus higher current efficiency of the process [26]. Bath temperature and liquidus temperature [27, 28] are important parameters with respect to the control of an electrolytic aluminium reduction cell because their evolution reflects the thermal balance [29]. The liquidus temperature gives information on bath composition [30]. Bath temperature and moreover, the difference between bath and liquidus temperature, superheat, determines current efficiency; too high a superheat leads to a decrease of current efficiency [31]. Although a low superheat results in higher current efficiency, lowering superheat decreases alumina solubility and alumina solution rates [32]. This may lead to sludge formation, resulting in horizontal current components in the metal pad [25]. The interaction of these current components with the magnetic field results in an unstable metal pad [33, 34]. Running a cell at too high a temperature also means that too much energy is wasted as thermal losses and anode carbon consumption increases [35]. If thermal cycling of the bath is reduced, thermal cycling of the cathode blocks is also reduced [36]. In addition, better control of superheat leads to more stable ledge formation [29]. Consequently, this may result in the benefit of increased cell lifetime [37]. The physical and chemical properties of the electrolyte impact cell design, electrolyte flow, cell life, productivity and power efficiency [7].

While molten cryolite is the main ingredient of the Hall-Heroult process, electrolyte additives are used to improve its chemical and physical properties. The main electrolyte additives used in reduction cell operation consist of various fluorides often added in the form of carbonates or oxides. These fluorides are primarily aluminium fluoride, AlF_3, alkali fluorides such as lithium fluoride, LiF, and alkaline earth fluorides including calcium fluoride, CaF_2, and magnesium fluoride, MgF_2 [27]. Sodium carbonate, Na_2CO_3, commonly called 'soda-ash', is also an important electrolyte additive. However, AlF_3 is the most commonly used bath additive in industrial reduction cells. AlF_3 is added to the electrolyte of the reduction cell to

lower the liquidus temperature and hence permit operation of the cell at temperatures below the melting point of pure cryolite [38]. The reduction in operating temperature is beneficial in that savings in power consumption and improvements in current efficiency can be realised [27]. AlF_3 is beneficial in reducing the solubility of metals in the electrolyte and for lowering bath density, both of which contribute towards improving the process efficiency [24]. AlF_3 content in the electrolyte is typically expressed as percent excess AlF_3 by weight (%wt), which is the weight percent of excess AlF_3 in the electrolyte so that pure cryolite has 0.0% excess AlF_3 [39]. The control of the weight ratio in the bath is determined by the rate of sodium absorption into the carbon lining of the cell, the sodium oxide content of the alumina, the bath temperature and the rate of AlF_3 evolution [39]. Further, AlF_3 is largely rejected from cryolite as it freezes into the ledge on the sidewall. For the duration of a reduction cell life, AlF_3 has to be added periodically in order to maintain a constant bath ratio.

Electricity - As the Hall-Heroult process operates at temperatures around 940.0 to 980.0°C then it is necessary to have electricity as an energy source to achieve these high temperatures. Direct current flow between the anodes and the liquid aluminium on the cathode surface drives the electrolysis process [23]. From the metal, current flows into the carbon cathode blocks and then into the steel collector bars embedded within the cathodes. The collector bars then carry the current out of the cell and link onto larger aluminium busbars which surround the cell. Subsequently, the busbars supply electrical current to the anodes of the next cell in the potline. Electrical current supply and aluminium production occur simultaneously and continuously in the electrolysis cell [23]. The electricity used to produce aluminium is relatively high. One kilogram of aluminium requires approximately three to four kilowatts of electrical energy [40].

Producing quality primary aluminium metal is the objective of the Hall-Heroult process and hence, is the only desirable output of the process. However, in producing aluminium there are certain unavoidable losses that occur as a result of the chemical reactions taking place in the reduction cell. While these losses, such as energy, in the form of heat, carbon monoxide and carbon dioxide, are byproducts of the process, it is not necessary to discuss further here the particularities of these losses other than to make the reader aware that they contribute to losses in operating efficiency.

An understanding of the aluminium production process is facilitated by consideration of a process flow chart for CABBL, as shown in Figure 2.1.9. Considering the aluminium production process presented and the particularities and characteristics of aluminium smelting it is evident that aluminium production industries are typically large scale, incorporating multi-stage, multi-variable and unstable operation in a highly dynamic environment. While intelligent sensors and analytical equipment have been developed in recent years to provide critical process information, continuous efforts have been made to develop detailed process models. Advanced process control has been considered in recent years in the aluminium industry as an opportunity to improve process efficiency. However, process non-linearity and insufficient understanding of the associated first principles of the Hall-Heroult process cause difficulties in developing such models. In addition, highly dimensional, multi-variable equations and statistical dependency among variables yield further problems in analytical modelling. Conceptually, first principles modelling is most appealing because it is based on applied mathematics, physics and chemistry. However, it has been noted that assumptions are generally required to simplify the model. Hence, while a considerable amount of time and expertise are required to develop an adequate model, the model may not work well in practice due to the simplifying assumptions made. As a result, statistical techniques offer an alternative in the aluminium industry for developing empirical models of the process [41, 42]. On the other hand, developments in neural network technology have created an alternative technique for the aluminium industry to develop reliable process models. Neural networks are a particular type of phenomenological modelling that use historical data to develop a model of a process. Neural networks offer a powerful mathematical technique for modelling, control and optimisation of dynamic processes that are developed using process data, without the need for understanding the associated scientific principles and underlying relationships among process parameters. While neural networks are being approved as a successful approach for model building, to predict and control specific parameters of many manufacturing processes, the investigations reported in the literature are generally of a conceptual theoretical nature. It is interesting to note that the practical application of neural networks for process prediction and control are very limited in the literature.

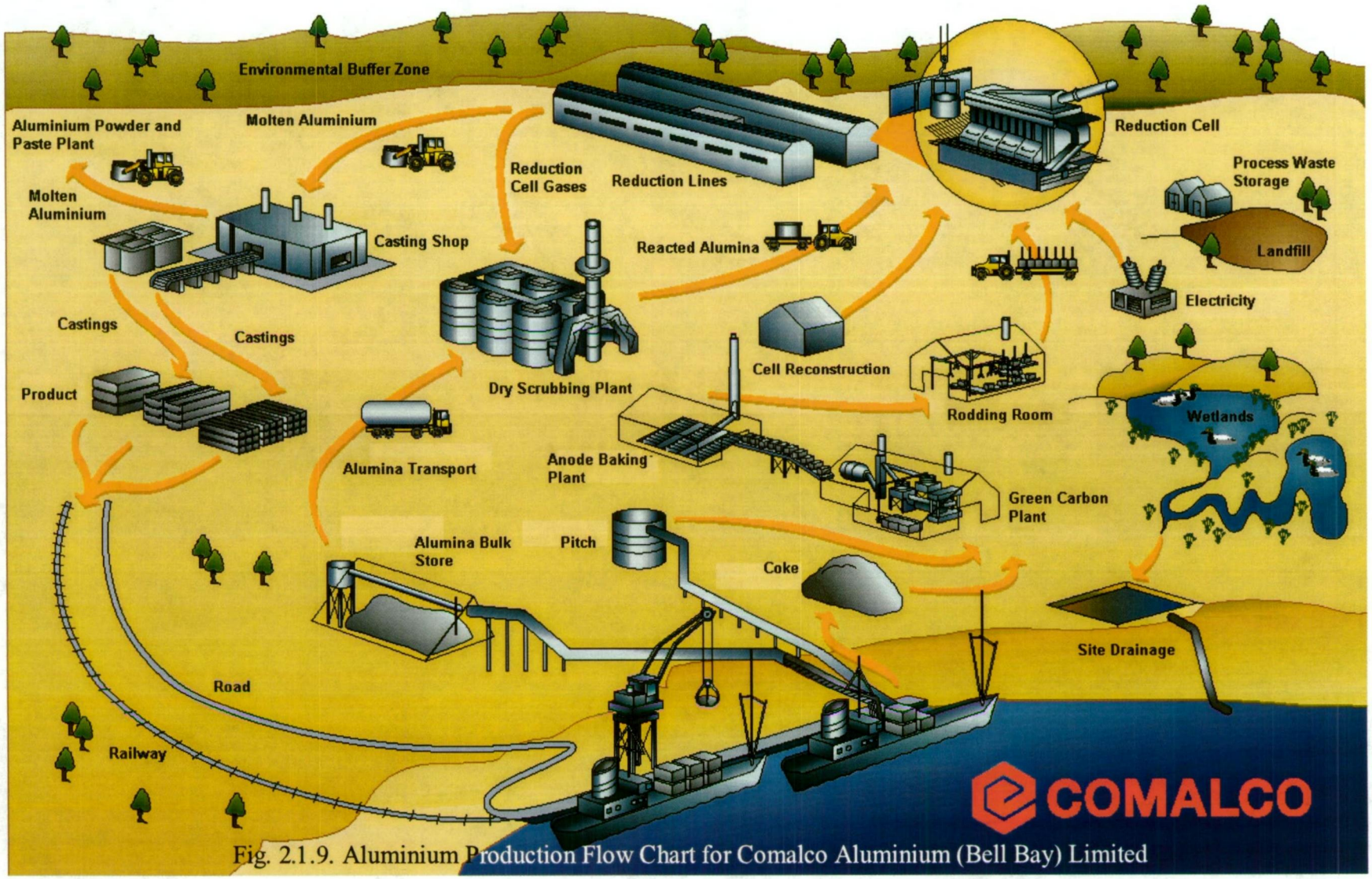


Fig. 2.1.9. Aluminium Production Flow Chart for Comalco Aluminium (Bell Bay) Limited

The following section introduces the reader to neural networks, providing a description of these computational tools, highlighting some of their applications and discussing in detail some of the more popular models of the neural network paradigm.

2.2 INTRODUCTION TO ARTIFICIAL NEURAL NETWORKS

A good starting definition of an artificial neural network, or more commonly, neural network, is that it is a type of information processing system whose architecture is inspired by the structure of biological neural systems [43]. A brief history of neural network development provides a useful introduction to these computational models with a comparison to biological neural systems following.

2.2.1 The Interesting History of Neural Networks

The year 1943 is often considered the initial year in the development of artificial neural systems [44]. It was in this year that McCulloch and Pitts [45] outlined the first formal model of an elementary computing neuron, which included all necessary elements to perform logic operations. While the McCulloch and Pitts model neuron was not feasible for implementation in the era it was developed, it laid important groundwork for future developments. Such developments came in 1949 when Donald Hebb [46] proposed a technique for updating neuron connections, now referred to as the 'Hebbian learning rule'. Hebb stated that information could be stored in connections and postulated the learning technique that had a profound impact on future developments in this field [44]. Further developments came in 1960 through work completed by Frank Rosenblatt [47] who focussed on the issue of developing weights between the connections of neurons for particular computational tasks. Rosenblatts' work concentrated on networks called 'perceptrons' in which the processing units were organised into layers with feedforward connections between one layer to the next [48]. For the simplest class of perceptrons without any intermediate layers, Rosenblatt was able to prove the convergence of a learning algorithm, a way to change the weights iteratively so that a desired computation was performed [48]. Around the same time Bernard Widrow and Marcian Hoff [49] invented a very similar device called 'adaline', and a new learning rule, called the 'delta rule', was developed. Early applications of adaline and its extensions to 'madaline', many adalines, include pattern recognition, weather forecasting and

adaptive control [48]. However, a publication by Marvin Minsky and Seymour Papert [50] in 1969, stating the limitations of perceptron-class networks, significantly slowed progress in the neural network area of computer science. However, during the period 1965 to 1984 attention was focussed on developing associative memory networks, in which different input patterns are associated with another if they are sufficiently similar. Prominent researchers in this field include Tuevo Kohonen [51] and James Anderson [52]. From 1982 to 1985 specific events caused a renewed interest in neural networks. John Hopfield [53, 54] completed some significant work in this time with the introduction of a recurrent neural network architecture for associative memories. With clarity and mathematical analysis, Hopfield showed how such networks could work and demonstrated their capabilities. In addition, a conference on competitive neural networks was held in Japan during this period in which Japanese researchers announced developments in the field of neural networks. United States periodicals printed this news, creating renewed funding for American researchers through fear of not developing similar technologies. Significant developments also came in 1986 through work completed by James McClelland and David Rumelhart [55] which reinspired the field of neural networks with the publication of new learning rules and other concepts which removed one of the most essential network training barriers that had grounded the mainstream efforts of the mid-1960s [44]. By 1986, with the publication of *Parallel Distributed Processing*, edited by Rumelhart and McClelland [56], the field of neural networks accelerated [57]. In 1987, the first open conference on neural networks in modern times, *Institute of Electrical and Electronic Engineers* (IEEE) *International Conference on Neural Networks*, was held with over 1,700 participants, and the *International Neural Network Society* (INNS) was founded [57]. Today, the IEEE and INNS along with other researchers continue to publish developments in the field of neural networks, highlighting improved learning rules, techniques for selecting optimum network parameters and new algorithms for developing optimum network architectures.

Although neurocomputing has had an interesting history, the field is still in its early stages of development [44]. So far, biologists and neurologists have concentrated their research on uncovering the properties of individual neurons. While the mechanisms for the production and transport of signals from one neuron to another are well understood physiological phenomena, how these individual systems

cooperate to form complex and massively parallel systems capable of incredible information processing feats has not yet been completely elucidated [58]. However, it is noted that neural networks are nonalgorithmic, nondigital and intensely parallel systems consisting of a number of very simple and highly interconnected processing units which are analogs of the biological neural cells in the brain [59].

2.2.2 Biological Neural Systems

By way of analogy it is useful to consider the basic concepts of biological neural systems, leading to an understanding of the structure of artificial neural networks. A sketch of a typical biological neural cell, or neuron, is shown in Figure 2.2.1 with the main features labelled.

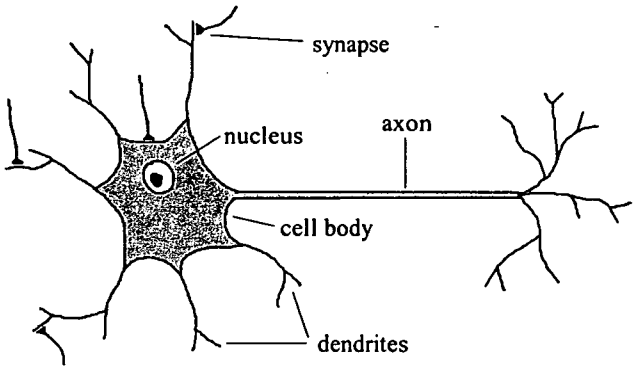


Fig. 2.2.1. Schematic Diagram of a Typical Biological Neuron

The main processing unit, cell body, accepts inputs from other similar neurons, processes these inputs and sends a single output to other neurons in the system. Each cell body has a single axon, a long cylindrical connection that carries impulses from the neuron, extending from it which joins with other neurons through connections called synapses. The synapse connections to other neurons occur on the many individual dendrites that are attached to each neuron in the system. Many, but not all, synapses are adaptive or plastic; that is, they can increase or decrease in strength under the proper conditions and as a result they can have differing strengths or synaptic weights [43]. The incoming impulses to a neuron can have either a positive or negative effect on the activation of that neuron and are referred to as excitatory or inhibitory respectively. For a neuron to produce an output its excitation must exceed its inhibition by an amount called the 'threshold' of the neuron. The frequency and

nature of the impulse transmitted by a neuron is largely determined by the synapses [60]. It is interesting to note that the human brain typically consists of approximately 10^{11} neurons with approximately 10^4 synapses per neuron [44].

2.2.3 Artificial Neural Systems

Artificial neurons are fundamental to artificial neural networks, forming the processing units for such systems. The artificial neuron is modelled on the basic concepts of the biological neuron. The first modelling of neurons dates back to the 1940s and was carried out by McCulloch and Pitts [45]. Drawing on their work on biological neurons, they proposed the following model [61]:

> "A synthetic neuron forms a weighted sum of the action potentials which arrive at it (each of these potentials is a numeric value which represents the state of the neuron which has emitted) and then activates itself depending on the value of this weighted sum. If this sum exceeds a certain threshold, the neuron is activated and transmits a response (in the form of an action potential) of which the value is the value of its activation. If the neuron is not activated it transmits nothing."

Similar to biological neurons, artificial neurons accept inputs from other similar neurons, process the inputs and send a single output to other neurons in the system. Likewise, artificial neurons communicate via weighted interconnects. The basic structure of an artificial neuron, shown in Figure 2.2.2, highlights the similarity between artificial and biological neurons.

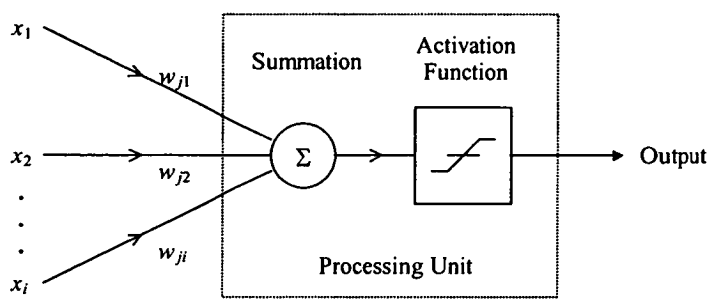


Fig. 2.2.2. Schematic Diagram of a Typical Artificial Neuron

The inputs to the neuron, shown as x_1 to x_i, represent dendritic connections in biological neurons, while the weighted connections, denoted as w_{j1} to w_{ji} , represent synapses. In addition, every artificial neuron has one output line, simulating the axon

of a biological neuron, which can branch out to form connections with other neurons. The cell body of a biological neuron is represented as a processing unit in the artificial neuron model. There are essentially three functions that combine to give the neuron its processing capability; the input function, activation function and output function. The input, activation and output functions of a neuron can usually be combined into one function, called the 'transfer function' [43]. To understand the transfer function it is useful to consider these three functions separately. The input function performs a summation of the multiplication of inputs with the corresponding weights. It is an additive function that essentially obtains the summed input for each neuron by multiplying the output of every neuron connected to it by the associated synaptic weight for each connection and then summing the result. The net input to the j^{th} unit, net_j, can be written as [62]:

$$net_j = \sum_i x_i w_{ji} \tag{2.2.1}$$

where, index i denotes the number of input neurons

The output from the input function forms the input for the activation function. The activation function is a non-linear function that, when applied to the net input of a neuron, determines the output of that neuron [63]. The activation function can be any function that is monotonically increasing and differentiable [64]. The values that the activation function can output is generally limited to the range 0.0 to 1.0 or -1.0 to 1.0, depending on the activation function used. Early neural models used a simple threshold function, or step function, as the activation function. This particular type of activation function allow a 1.0 to be output from the neuron if the weighted sum of the inputs exceeds some threshold, otherwise the output is 0.0. The threshold function is shown graphically in Figure 2.2.3(a). More recently the threshold function has been replaced by a more general non-linear sigmoid function, also called the logistic function [48]. A sigmoid function may be loosely defined as a continuous, real-valued function whose derivative is always positive and whose range is bounded [63]. The logistic function is written as [65]:

$$f(net_j) = \frac{1}{1 + \exp(-net_j)} \tag{2.2.2}$$

A graphical representation of the logistic function, shown in Figure 2.2.3(b), highlights the bounded range of the function, 0.0 to 1.0.

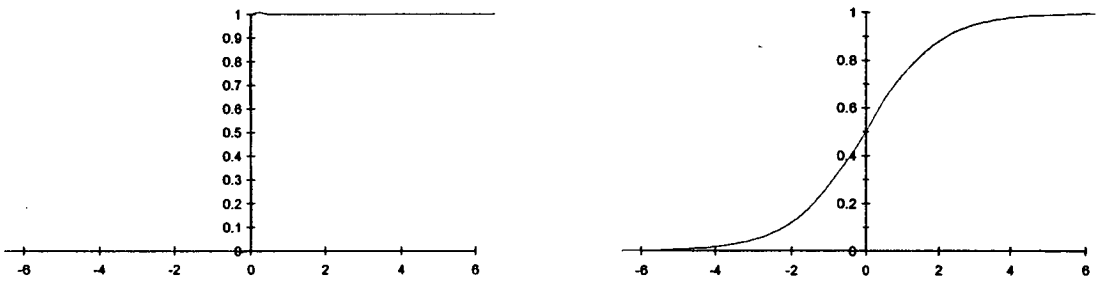


Fig. 2.2.3. Graphical Representation of (a) Threshold Function, and (b) Logistic Function

An advantage of this particular type of function is that its derivative, which will later be shown to be a significant aspect of neural computation, is easily calculated. In fact the derivative of the logistic function, $f'(net)$, is written as [66]:

$$f'(net_j) = f(net_j)\,(1 - f(net_j)) \tag{2.2.3}$$

The third component of the transfer function, the output function, is usually chosen to be equal to the output of the activation function, ie. the output of the neuron will be the same as the activation [43].

One of the simplest forms of a neural network is the perceptron, developed by Frank Rosenblatt in the early 1960s. The perceptron is a pattern classification system that recognises abstract and geometric patterns from optical input patterns, despite noise in the input [67]. The computation that takes place in the processing neurons of the perceptron is based on a simple principle. The neuron computes the weighted sum of the input signals and compares that net weighted input to a threshold value, T [59]. If the net input is greater than or equal to the threshold, the neuron outputs a value of

1.0, otherwise it outputs a value of -1.0 [59]. Hence, the transfer function used in the hidden and output layer neurons of the perceptron is written as [59]:

$$y = \begin{pmatrix} +1, \text{if } I \geq T \\ -1, \text{ if } I < T \end{pmatrix} \tag{2.2.4}$$

where, y = output of the neuron,

I = net weighted input to the neuron and is calculated as follows:

$$I = \sum_{i=1}^{n} w_i x_i \tag{2.2.5}$$

where, w_i = component of weight vector, and

x_i = component of input vector

Rosenblatt introduced a training algorithm for the perceptron that provided the first procedure that could be used to allow a network to learn a task [59]. This training algorithm was used for changing weights in the network using the formula [59]:

$$w_{new} = w_{old} + \beta\, y\, x \tag{2.2.6}$$

where, w_{new} = new weight vector,

w_{old} = old weight vector,

$$\beta = \begin{pmatrix} +1, \text{if the perceptrons answer is correct} \\ -1, \text{ if the perceptrons answer is wrong} \end{pmatrix}$$

y = perceptron output, and

x = input vector

Frank Rosenblatt and other researchers were able to demonstrate mathematically that the perceptron training algorithm can always solve any linearly separable problem in a finite number of steps [43]. The perceptron learning rule convergence theorem states that if weights exist to allow the network to respond correctly to all training patterns then the rule's procedure for adjusting the weights will find values such that the network responds correctly to all training patterns, ie. the network solves the

pattern or learns the classification [68]. However, the perceptron never became a viable application system due to certain limitations associated with its ability. Nevertheless, the development of the perceptron laid some important groundwork and inspiration for other researchers to further develop neural networks to the stage they are today and therefore is seen as a historical landmark in the field of neural systems.

Neural networks are trained for particular applications such that they learn to solve associated problems, they are not programmed to do so. Hence, 'training' is fundamental to all neural networks and is the process of modifying the network interconnection weights. Successful training results in a real number being assigned to every interconnection in the network such that every interconnection has a unique associated weight and the performance accuracy of the network is maximum. By varying the weights associated with each input, the neural network can, in conjunction with the transfer function, implement any transformation between its inputs and outputs [69]. However, these weights need to be computed for each particular application and because it is not usually possible to compute them directly, this is achieved by the repetitive and often time-consuming process called 'neural network training' [69]. Algorithms for varying these connection strengths or weights such that learning ensues are called 'learning rules' [70]. The specific algorithms used for learning are dependent largely on the particular network model being considered, hence, the respective algorithms are discussed in relation to the network that they apply in later sections. Learning methods for neural networks may be broadly grouped as supervised and unsupervised, with a great many paradigms implementing each method [71]. Prior to discussing these two particular training techniques, however, it is useful to consider a taxonomy of neural networks, as shown in Figure 2.2.4. Neural networks, in addition to being classified by their training technique, either supervised or unsupervised, are classified as either feedforward or recurrent networks, as the respective categories reflect the processing behaviour of the network. Hence, a discussion of the difference between feedforward and recurrent network models is useful, followed by a discussion of the training techniques.

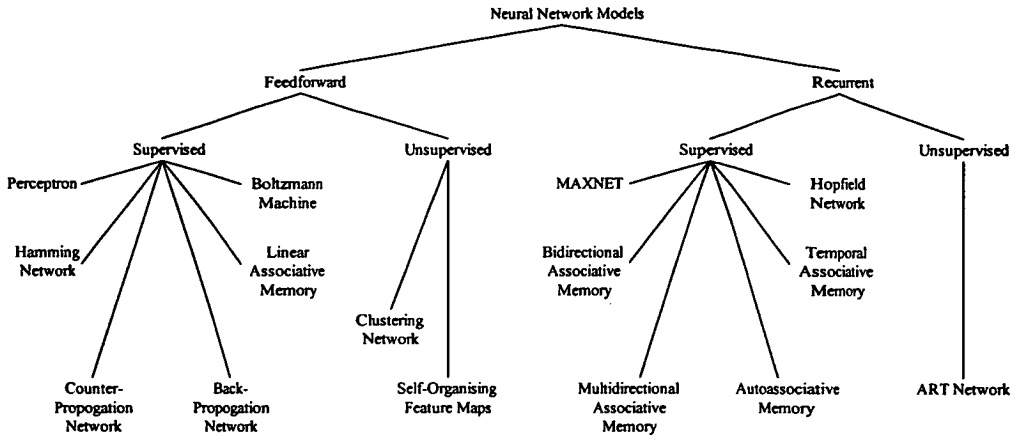


Fig. 2.2.4. Taxonomy of Neural Network Networks Classified as Feedforward or Recurrent and Supervised or Unsupervised [72]

Feedforward Networks - In a feedforward network, input activity signals propagate in one direction only, from the input stage through intermediate neurons to an output stage. The basic feedforward network contains three distinct layer types; input, hidden and output. This type of neural network has one input layer, one output layer and any number of hidden layers in between [55, 73]. The basic architecture of a feedforward network is shown in Figure 2.2.5 with the main features labelled.

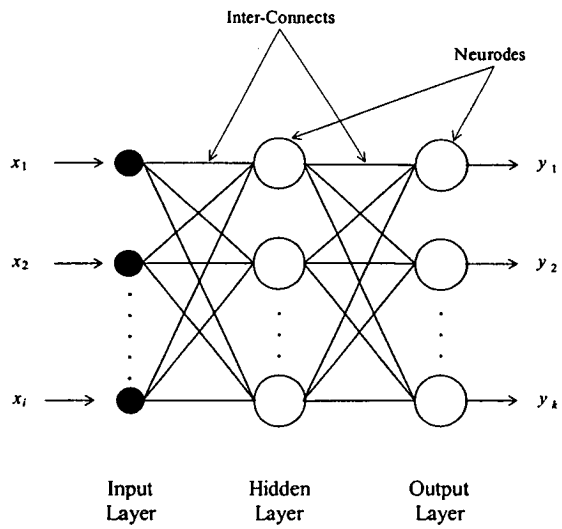


Fig. 2.2.5. Basic Structure of a Multi-Layer Feedforward Neural Network [48]

Each layer in the network contains neurons which receive any number of inputs, process those inputs and send a single output to other neurons in the subsequent layer. The neurons in the input layer receive information from a knowledge base

while the output layer neurons send information to the surrounding environment. The neurons in the hidden layer are the processing units of the network. A classic example of a feedforward neural network is the multi-layer perceptron [48, 60, 74].

Recurrent Networks - A recurrent neural network distinguishes itself from a feedforward neural network in that it has at least one feedback loop [75]. For example, a recurrent network may consist of a single layer of neurons with each neuron feeding its output signal back to the inputs of all other neurons, as illustrated in the architectural graph of Figure 2.2.6. Hence, due to the existence of feedback connections among neurons, a recurrent neural network exhibits dynamic behaviour. Specifically, given the initial state, the state of a recurrent neural network evolves as time elapses. If the recurrent neural network is stable, a state of equilibrium can eventually be reached [76]. Recurrent neural networks are usually used for storing information as associative memories and solving computationally intensive problems. A classic example of a recurrent network that learns through supervised training is the Hopfield network, introduced by John Hopfield [53, 54].

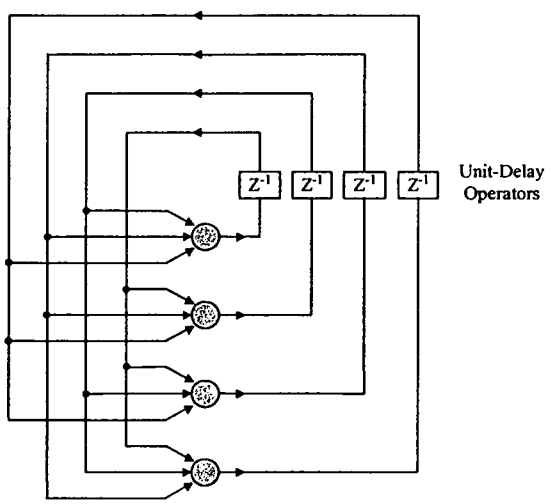


Fig. 2.2.6. Basic Structure of a Recurrent Neural Network with Feedback Loop [75]

Supervised Training - During supervised training a neural network adjusts its weight vector in the direction that minimises the error between its outputs and the targets to be learned [77]. Supervised training involves a teacher providing input patterns to the network that are expected to be encountered during operation and an associated output pattern that the network is expected to produce when it receives the particular

input pattern. An iterative process is used in conjunction with this style of learning such that the network adjusts its weights continually until the error between the network output prediction and the target output is minimum. Once the minimum error is achieved the network weights remain unchanged and the process is considered to have converged. The process of updating the network weights once is referred to as an 'epoch', or 'iteration', while the number of input-output patterns used for the training set is referred to as the 'epoch size'. Some researchers use an epoch size of one, meaning that the weights are updated after each training case is presented, however, it is usually preferred to use the entire training set for each epoch as this favours stability in convergence to the optimal weights [63].

Unsupervised Training - Unsupervised training differs from supervised training in that it only requires input vectors to train the network. During unsupervised training, network weights are adjusted so that similar inputs produce similar outputs [71]. For unsupervised training, the only available information is in the correlations of input data or signals. The network is expected to create categories from these correlations and to produce output signals corresponding to the input category [48]. The system searches for similar features in the training inputs to group them into categories where the numbers of a single category share common features [78]. Unsupervised learning algorithms use patterns that are typically redundant raw data having no labels regarding their class membership or associations and in this mode of learning the network must discover for itself any possibly existing patterns, regularities or separating properties [44]. In discovering these the network undergoes changes in its parameters, hence the name self-organisation often associated with these particular network models [79].

In addition to being either feedforward or recurrent and the technique adopted to train the network, there are many properties and characteristics that distinguish one neural network from another. Moreover, an artificial neural network is generally defined by [80]:

i). network properties - network topology, types of connections, order of connections and weight range

ii). node properties - activation range and transfer functions, and

iii). system dynamics - weight initialisation scheme, activation formula and learning rule

It is convenient to visualise neurons as arranged in layers, where typically, neurons in the same layer behave in the same manner [68]. That is, the factors that determine the behaviour of the neuron, namely its activation function and the pattern of weighted connections over which it sends and receives signals, are the same within each layer. The arrangement of neurons into layers and the connection patterns within and between layers defines the network architecture, which is commonly either single-layer or multi-layer. Typically, the input layer of neurons is not included in the determination of the network layer size as the input layer performs no computation. It simply passes forward the input pattern presented to the network to neurons in the subsequent layer. Hence, a single-layer network consists only of an input and output layer while a multi-layer network has in addition any number of hidden layers between the input and output layers.

2.2.4 Important Techniques for Improving Neural Network Performance

There are many important considerations when designing the neural network training and testing data and architecture. In order to achieve optimum performance from the network there are some decisions to be made in regard to the data that is used to train the network and the preparation of that data, as well as the particular network architecture to use. In addition, subsequent testing of the network after training is complete is necessary to evaluate neural network performance. In designing the training data for neural networks there are a number of important considerations for developing a satisfactory training set. The performance of a network depends heavily upon the vectors used to train it [71]. One of the most significant considerations is selecting the size of the training set. If the neural network is going to be effective at its ultimate task, the training set must be complete enough to satisfy two goals [63]:

i). Every variable in the training data set must be adequately represented. Usually, the training data will consist of several possible subgroups, each having its own central tendency toward a particular pattern. All of these patterns must be represented sufficiently.

ii). Within each class, statistical variation must be adequately represented. It is the presence of random noise imposed onto pure patterns that makes most neural network applications necessary. The training set must be designed to insure that an adequate variety of noise effects are included.

Neural network performance can often be improved if the data set is modified by removing insignificant characteristics. The important aspects of the training data are often independent of the value of offsets and standard deviations, these may merely obscure the issue and complicate the networks task [71]. Scaling the network input, or 'normalising', is one such method of removing insignificant characteristics from the training set by scaling the magnitude of each input vector component between some predetermined limits. Uniform scaling of the input data results in the individual components of the input vector being recognised by the network to be of equal magnitude. Consequently, the network is not bias towards components in the input vector that are of a higher magnitude. Further, normalisation of target output data used for training the network is recommended as most training algorithms minimise the total error of all outputs. If target output variables are unequally scaled, those with larger variabilities will be favoured, as they will dominate the error sum [63]. Normalisation of target output data is mandatory when the output of the network is bounded due to the particular activation function used. One specific example of this is the sigmoidal function whose output has been shown to be limited to the range 0.0 to 1.0. For this particular function the output target values that the network is learning from must also be scaled to the bounded range to allow the network to learn the data. A common method of normalising data between the range 0.0 to 1.0 for a particular set of values of any magnitude is using the following formula:

$$norm(x_i) = \frac{x_i - \min(x_j)}{\max(x_j) - \min(x_j)} \tag{2.2.7}$$

where, $norm(x_i)$ = normalised i^{th} value in a set of j values,

x_i = original i^{th} value in a set of j values,

$\min(x_j)$ = original minimum value in a set of j values, and

$\max(x_j)$ = original maximum value in a set of j values

Through use of this method the data is scaled over the range 0.0 to 1.0 such that when x_i equals $\min(x_j)$, $norm(x_i)$ equals 0.0, and when x_i equals $\max(x_j)$, $norm(x_i)$ equals 1.0. Consequently, the remainder of x_i values between $\min(x_j)$ and $\max(x_j)$ are scaled uniformly between the bounded range 0.0 to 1.0.

While the training set is used to train the network, the test set is used to assess the performance of the network after training is complete [81]. For neural learning to be successful, it is essential for the system to perform correct classification of test samples on which the system has not been trained [82]. When training is complete it is interesting to try patterns not in the training set to see whether the network can successfully generalise what it has learned [48]. Generalisation in neural networks may be viewed as multi-dimensional interpolation [71]. To understand this, it is useful to consider a one dimensional input vector. For a given set of samples in the training set the individual values form some underlying, or unknown, curve. After training is complete the presentation of an input vector not included in the training set will require the network to generalise. Hence, it is necessary for the network to interpolate between points at the extremes of the input vector group so that given an intermediate value, x, the network can determine output values, $y = f(x)$, that lie on the unknown curve [57].

One of the most important attributes of layered neural network design is choosing the network architecture [44]. The network architecture is a very important consideration for the optimal trainability and generalisation ability [83]. For feedforward network models this decision involves the selection of how many hidden layers are necessary and how many neurons are required within each hidden layer. Generally, the number of neurons in the input layer is equal to the dimension of the input vector to be classified, generalised or associated with a certain output quantity. Similarly, the number of neurons in the output layer is equal to the number of required outputs. For example, the number of required outputs could be the number of possible classifications for a given set of inputs or the number of parameters to be predicted by a network. However, the decision of how many hidden layers and hidden layer neurons is more complex. For feedforward networks, it has been proven that there is no theoretical reason to ever use more than two hidden layers [63] and for the vast majority of practical problems it is rarely necessary to use more than one hidden layer

[71]. The use of two hidden layers is usually only necessary in practice when the network must learn a function that has discontinuities [63]. The decision of how many hidden layers to use is quite critical as the use of a second unnecessary hidden layer can dramatically slow network training. This is due to two effects [63]:

i). The additional layer through which errors must be backpropagated makes the gradient more unstable. The success of any gradient-directed optimisation algorithm is dependent on the degree to which the gradient remains unchanged as the parameters change.

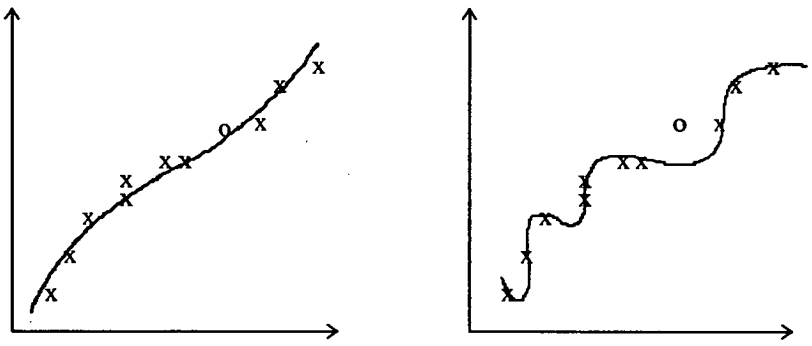
ii). The number of false minima usually increase dramatically, meaning that there is a higher probability that after many time-consuming iterations, the network will be stuck in a local minima, resulting in the need to restart training.

It is strongly recommended that one hidden layer be the first choice for any practical feedforward network design and if using a large number of hidden layer neurons does not satisfactorily solve the problem, then it may be worth trying a second hidden layer and possibly reducing the total number of hidden layer neurons [63].

Choosing an appropriate number of hidden layer neurons is extremely important as using too few will starve the network of the resources it needs to solve the problem, while using too many will increase the training time and may cause a problem known as overfitting [63]. Overfitting is illustrated in Figure 2.2.7, which demonstrates a curve fitted to a number of data points. In Figure 2.2.7(a) a good fit to the training data set is demonstrated, while in Figure 2.2.7(b) overfitting has occurred, as may be the case if too many hidden layer neurons are used. The circle in the figure represents a test set that the network may encounter. For a good fit, reflecting only a minimum number of hidden layer neurons interpolation is reasonable, however, for overfitting, reflecting the use of too many hidden layer neurons, interpolation is poor.

A guideline for selecting the optimum number of hidden layer neurons is to use as few as possible to obtain a satisfactory solution. A common method for determining the minimum number of hidden layer neurons required is to compare the error of the network for a range of neurons. The minimum number of neurons that can be used without increasing the associated network error represents the number of neurons that

should be used in the final model. This technique is discussed further in a later section, as appropriate.



(a) (b)

Fig. 2.2.7. Illustration of (a) Good Fit to Noisy Data, and (b) Overfitting of the Same Data [48]

2.2.5 Applications of Neural Networks

Since the late 1970's knowledge based systems have been used in various manufacturing domains [84]. However, they are less effective in the ever-changing, complex and open system environment of today's manufacturing processes [85]. On the other hand, neural networks emerged as a revolutionary technology for solving problems that are difficult for traditional computation [72]. Some specific advantages of neural networks that make them a desirable choice for a variety of applications are automated learning from examples, no need to make assumptions about the form of the relationship between input and output, and relatively fast learning [86]. However, while the advantages are noted it is also necessary to realise that there are also some specific disadvantages associated with neural networks. In particular, it is difficult to interpret the neural network solution and it is difficult to incorporate knowledge of a given problem into a neural network model. This combination of advantages and disadvantages make neural networks particularly useful in situations where sufficient training examples are available but there is no clear relationship between input and output [86].

Applications of neural networks are emerging at an ever increasing rate. Artificial neural networks have shown great performance in many fields and have proven ability in solving problems difficult to solve using conventional methods of

computation [87]. Consequently, neural networks are continually progressing and their applications are wide spreading [88]. A few of the areas where neural networks are finding applications include:

i). quality prediction in industrial control [89],
ii). manufacturing applications, such as design, process planning and scheduling [90, 91],
iii). process control, fault diagnosis and condition monitoring [92-95],
iv). control and monitoring of machining processes [96-99],
v). pattern recognition [100-102], and
vi). robotic control [103, 104].

While the domain of neural network applications are given briefly here it is the application of neural networks to industrial process control that is the main focus in this instance. However, it is important to note that most of the work documented here is in its theoretical concept; practical use and application of neural networks to industrial processes is extremely limited in the available literature.

2.2.6 Applications of Neural Networks for Industrial Process Control

Since neural networks act as universal function approximators they are expected to be very valuable in control applications, in particular, in situations where the plant to be controlled is not itself easy to model [105]. One particular application is where a neural network is trained to become a model of the process or plant. For this application, the network input and output training sets are developed using historical process data [105]. In this way a neural network model is developed that can be used as a controller for a process. Process modelling and control are a manufacturing domain where neural networks can play a very important role [72]. The use of neural networks in control applications - including process control, robotics, industrial manufacturing and aerospace applications, among others - has recently experienced rapid growth [106-109]. There is a distinction between a neural network model of a system and a neural controller for a system. Often a neural controller will make use of a system model either for training or operation, or both. In the case of one step ahead control, a neural model is used as control decisions may be made on the strength of the next predicted system state [110].

In the process industry, usually human operators rather than mathematically based advanced algorithms are used to achieve product control [72]. Gingrich [111] and Jalel *et al* [112] discussed a methodology that uses neural networks for capturing the knowledge of human process operators. The results show that for simple systems it is possible to learn the control actions of a human operator using neural networks. The neural network approach is preferred over heuristic approaches as there is no need for the operator to formulate his knowledge as rules and to train a neural network is easier than to design, build, and maintain an expert system [72]. Further, the advantages of using neural network approaches in process modelling and control are indicated by Chryssolouris and Guillot [113]. Chryssolouris and Guillot combined process modelling and artificial intelligence techniques for the determination of the operational range of process parameters. The process parameters were determined by neural network learning techniques. The authors pointed out that while synthesis of multiple sensor information would provide better results, neural networks excelled in dealing with situations in which process models do not adequately reflect the process complexity. A neural network's ability to learn a control algorithm without *a priori* analysis or modelling can be of significant benefit for difficult, complex, and non-linear control applications [72]. In addition, Willis *et al* [89] have shown the possibility of using neural network models directly within a model based control strategy, making use of an on-line optimisation routine to determine the optimal settings for standard industrial controllers. Application of the control algorithm to a non-linear distillation system was used to indicate the potential of the neural network based control philosophy. On the other hand, Madey *et al* [114] used a neural network simulator to control a production system which took inputs relating to raw material density, machine feed rate and machine feed pressure and produced outputs relating to the probability of machine breakdown, the production rate and the number of defective units produced. In addition, the outputs provide a measure of profit. The task for the network was to predict the required settings that produced the maximum profit, given the current settings and production rates.

While there are some reports of neural networks applied to modelling, control and optimisation of processes there is a need to select the most appropriate neural network for a specific application. While there are many networks that are capable of modelling a particular application, there will typically exist at least one network that

will perform better than its competitors for a given task. As neural networks are an advanced control technique that are often used as an opportunity to maximise corporate revenue, it becomes necessary to develop a set of selection criteria for selecting an artificial neural network that produces optimum performance for a specific application. The particular network selected is then used in preference to other models, for reasons including; it produces more accurate results, requires less information to develop the model or develops a model of the process in a shorter period. While these selection criteria are discussed later, it is useful now to introduce and discuss some particular models of the neural network paradigm that are used as an integral part of the research being completed in this instance.

2.3 PARTICULAR MODELS OF THE NEURAL NETWORK PARADIGM

In this section the specific details of the particular network models that are studied and applied as a part of this research project are given. It is important to note that while the associated architecture, algorithms and particularities of the studied network models are provided here, the author has translated the appropriate algorithms for each network model into Pascal programming code, such that the capabilities, operation and behaviour of each of the studied neural network models can be investigated. The neural network programs listed in the following are included in the accompanying software. In addition, the associated programming code is included in formatted descriptive form on the accompanying software, while a detailed user guide for the neural network programs is attached as Appendix A.

- Widrow-Hoff (WH) Neural Network
- Backpropagation - 1 hidden layer - (BP1) Neural Network
- Backpropagation - 2 hidden layers - (BP2) Neural Network
- Radial Basis Function (RBF) Neural Network
- Kohonen (KOH) Neural Network
- Radial Basis Function - incorporating Kohonen - (RBFKOH) Neural Network
- General Regression Neural Network (GRNN)

The particular models listed here are feedforward type networks that learn using a supervised training technique, with the exception of the Kohonen network, which incorporates the unsupervised training methodology. While the majority of

documented manufacturing applications use the backpropagation network, due to its proven success in many areas of manufacturing, alternative models such as the Widrow-Hoff, radial basis function and general regression neural networks were selected in order to provide a comparison with the more popular backpropagation model. In addition, the Kohonen network is used in conjunction with the radial basis function network to cluster the inputs in an attempt to improve the radial basis function network performance. The operating mode and characteristics of the Kohonen network make it suitable for this particular task and hence, selection as a model for this particular research project. In addition, considering typical applications in the aluminium industry involve process control, prediction and optimisation, the particular models of the neural network paradigm listed here are most appropriate for modelling the majority of applications. The Widrow-Hoff, backpropagation and radial basis function neural networks perform well and have proven documented success with continuous mapping applications and were chosen because of this. Further, the general regression neural network was selected due to its modelling capability, in addition to the fast training times available with this particular network. Nevertheless, it is useful to note that if the selected neural network models are found to be inadequate for the applications encountered in the aluminium industry then further models will be developed as required.

2.3.1 Widrow-Hoff Neural Network

The Widrow-Hoff (WH) neural network was introduced by Widrow and Hoff [49] in the early 1960s and is an adaptive patten-classification machine. This type of network learns through supervised training, using a learning law called the 'delta rule'. This learning rule minimises the mean squared error between the activation and the target value. During training, the adjustable weights connecting the input units to the output unit are updated to minimise the error between the predicted output and the target output supplied to the network. The delta rule for adjusting the i^{th} weight, w_i, for each activation pattern, is written as [68]:

$$\Delta w_i = \alpha(t - y\_in)x_i \tag{2.3.1}$$

where, α = learning rate,

t = target output,

y_in = net input to output unit y and is given by:

$$y\_in = \sum_{i=1}^{n} x_i w_i \text{, and} \tag{2.3.2}$$

x_i = vector of activations of input units

Hence, the objective of the delta rule is to find an optimum set of synaptic weights, w_1, w_2, ..., w_i, that minimise the error between the desired target and the corresponding network output.

Architecture - In its simplest form, this network consists of a single neuron along with its associated input interconnects and synapses [43]. The architecture of the WH network is shown in Figure 2.3.1. The processing element of the WH model is a single unit that is capable of receiving input from several units. A bias unit is also necessary, hence, an input with a constant value of +1.0 is supplied to the network with each activation, connected to the output unit through a weighted connection, *b*. The basic WH neural network has one output unit and *i* input units. However, several processing elements that receive signals from the same input units can be combined in a single layer network.

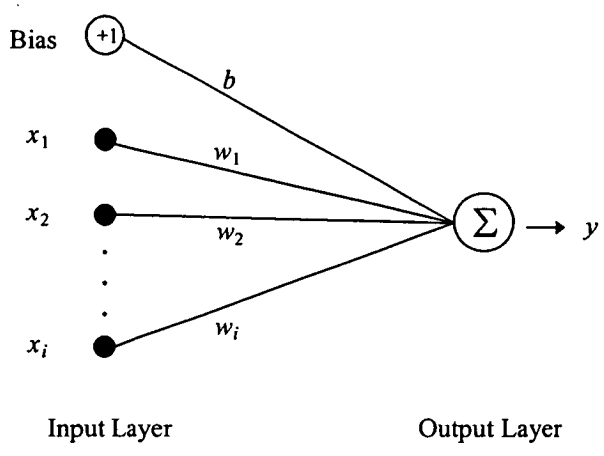


Fig. 2.3.1. Basic Architecture of WH Neural Network

Algorithm - A training algorithm for the WH neural network is written as follows [94]:

Step 1. Initialise network weights, w_i, to small random values.
Set learning rate, α.

Step 2. While stopping condition is false, do Step 3.

Step 3. For each bipolar training pair, $s:t$, do Steps 4 to 7.

Step 4. Set activations of input units using

$$x_i = s_i \tag{2.3.3}$$

Step 5. Compute net input to output unit y, where:

$$y\_in = b + \sum_i x_i w_i \tag{2.3.4}$$

Step 6. Update bias, b, and weights, w_i, using:

$$b(new) = b(old) + \alpha(t - y\_in), \text{ and} \tag{2.3.5}$$

$$w_i(new) = w_i(old) + \alpha(t - y\_in)x_i \tag{2.3.6}$$

Step 7. Test for stopping condition. If the largest weight change that occurred in Step 3 is smaller than a specified tolerance, then stop, otherwise continue.

In order to determine the success and sufficiency of supervised training using the delta rule it is necessary to have a quantitative measure of learning. As this supervised training algorithm involves the reduction of an error value then it follows that an error value be used to evaluate network training. Root-mean-squared, RMS, error is an adequate and commonly used error measure and is computed using the following formula [81]:

$$\text{RMS error} = \sqrt{\frac{\sum_p \sum_k (a_{kp} - t_{kp})^2}{n_p n_k}} \tag{2.3.7}$$

where, p = number of data patterns,

k = number of output neurons,

a_{kp} = the output value produced by output neuron k after presentation of pattern p,

t_{kp} = target output for output neuron k after presentation of pattern p,

n_p = number of patterns in the data set, and

n_k = number of neurons in the output layer

The RMS error is a useful measure of how close a network is getting its predictions to its target output values. For successful training the RMS error will decrease significantly in the initial stages of training and converge after a sufficient number of iterations have been completed. Generally, a RMS error value less than 0.1 indicates that a network has sufficiently learned its training set [81].

Applications - Although the WH neural network is a relatively simple model it has proven success in a variety of applications, including finance and investment [115] and weather forecasting [76].

2.3.2 Backpropagation Neural Network

The backpropagation (BP) model is the most widely used of the neural network paradigms and has been applied successfully in a broad range of areas [74, 103, 91, 99]. BP networks are multi-layered feedforward neural networks that are trained using the error BP procedure, a supervised mode of training. BP is a systematic method for training multi-layered artificial neural networks and is a form of supervised training [77].

Architecture - The architecture of a BP network, as shown in Figure 2.3.2, consists of an input layer, one or more hidden layers and an output layer. There are i input nodes, j hidden nodes and k output nodes. All input nodes are connected to all hidden nodes

through weighted connections, w_{ji}, and all hidden nodes are connected to all output nodes through weighted connections, w_{kj}.

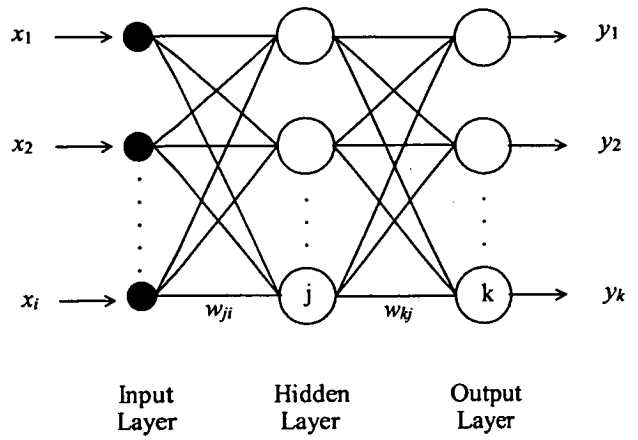


Fig. 2.3.2. Basic Structure of BP Neural Network [48]

The sole function of the input neurons is to pass forward input patterns to neurons in the hidden layer. In this type of feedforward network there are no connections leading from a unit to units in previous layers, nor to other units in the same layer, nor to units more than one layer ahead [48]. Hence, every neuron in each layer communicates only with neurons in the immediate following layer. Only the hidden layer and output layer neurons complete any type of processing. The neurons in these layers perform three functions; an input function, an activation function and an output function. The input function performs a summing of the inputs and synaptic weights. It is a linear function and is given by the equation:

$$net_j = \sum_j x_i w_{ji} \tag{2.3.8}$$

where, net_j = weighted summed input to neuron j,

x_i = input i to neuron j, and

w_{ji} = weight connecting input neuron i to hidden layer neuron j,

The most common activation function used in BP networks is the logistic function, which is a non-linear sigmoidal function given by the following equation:

$$f(net_j) = \frac{1}{1+\exp(-net_j)} \tag{2.3.9}$$

While it is shown that the input function summation is used in the processing performed by the activation function, it is noted that the purpose of the output function is to pass forward the output of the activation function. Hence, the output function is a linear function, it is equal to the output of the activation function.

Each training pattern presented to a BP network is processed in two stages. In the first stage, the input pattern presented to the network generates a forward flow of activation from the input to the output layer. In the second stage, error in the network output generates a flow of information from the output layer backward to the input layer [43]. The error BP procedure uses a gradient descent method which adjusts the weight in its original and simplest form by an amount proportional to the partial derivative of the error function with respect to the given weight [72]. The calculation of associated error for a given input pattern is determined only after the forward propagation of the pattern is complete. Specifically, for each neuron in the output layer of the BP network a single real number is output, which is compared to a target value presented with each input pattern. The error associated with this comparison is then used to update the weights for all interconnections from the hidden layer to the output layer. Similarly, an error value is calculated for all neurons in the hidden layer immediately prior to the output layer and subsequently, all weights are updated for interconnections that form inputs to this hidden layer. This process is completed until the last layer of weights has been updated in this manner. The error value, δ, is simple to compute for the output layer and somewhat more complicated for the hidden layers [81]. Considering firstly the output layer, the error value, δ_k, is calculated as follows [81]:

$$\delta_k = (t_k - a_k) f'(net_j) \tag{2.3.10}$$

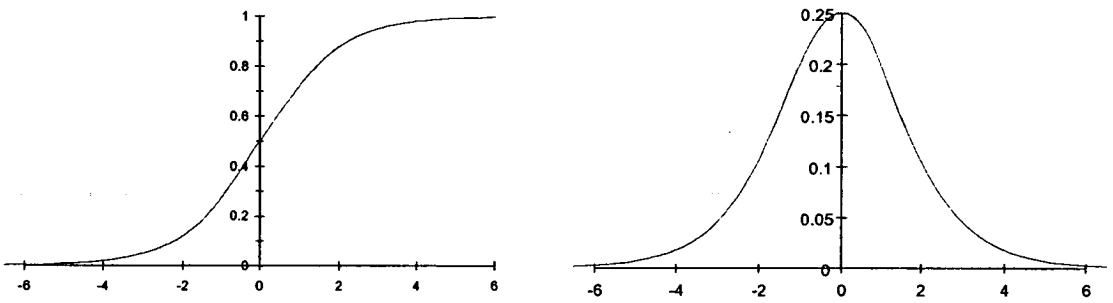
where, t_k = target value for unit k

a_k = output value for unit k

$f'(x)$ = derivative of the sigmoid function, and

net_j = weighted sum of inputs to hidden layer neuron j

Hence, the quantity (t_k - a_k) represents the difference between the target output and the network prediction while the derivative of the sigmoid function is used to scale the error. For the commonly used sigmoid function the maximum value of the derivative corresponds to the point of maximum slope on the function curve. It is useful to reproduce a graph of the logistic function together with a graph of the derivative of this function, as shown in Figure 2.3.3, in order to highlight this important relationship. It can be seen that by including the derivative term in the error value calculation the error is scaled to make a larger correction when the weighted sum of the inputs is small, close to zero, and a smaller correction when the weighted sum of the inputs is large.



(a) (b)

Fig. 2.3.3. Graphical Representation of (a) Logistic Function, and (b) Derivative of Logistic Function

For hidden layer neurons, the associated input and output weighted connections are shown in Figure 2.3.4 to illustrate the weights that are to be considered in the weight update for hidden layer neurons.

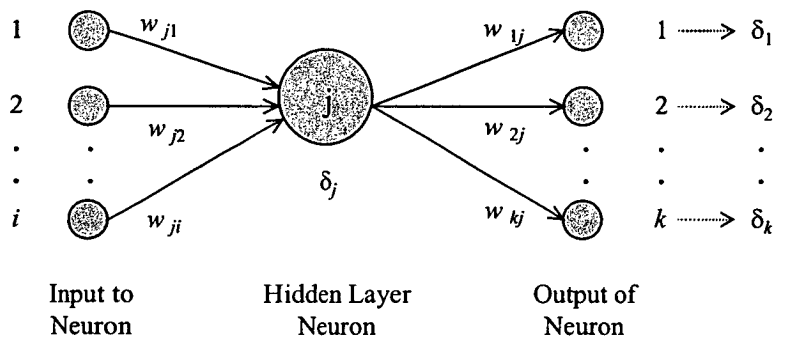


Fig. 2.3.4. Illustration of Hidden Layer Processing Neuron Used in BP Neural Network [81]

For neuron j in the hidden layer the error value calculation considers the weighted sum of the δ values of all neurons that receive output from neuron j. Hence, the error value calculation for the hidden layer is written as [81]:

$$\delta_j = \left[\sum_k \delta_k w_{kj}\right] f'(net_j) \tag{2.3.11}$$

where, w_{kj} = weight connection to neuron k from neuron j

The weight adjustment for the interconnections of the output and hidden layer neurons is now calculated using the respective δ values. Therefore, each interconnection weight is adjusted by considering the δ value of the neuron that receives input from that interconnection. Hence, the interconnection weight adjustment is written as [81]:

$$\Delta w_{ji} = \eta\, \delta_j\, a_j \tag{2.3.12}$$

where, w_{ji} = weight connection to neuron j from neuron i, and

η = learning rate constant, $0 < \eta < 1$, and

a_j = output of hidden layer neuron j

It can be seen that updating of interconnection weights is primarily based on three parameters, η, δ_j and a_j. As Δw_{ji} is proportional to δ_j then it follows that a large error value from neuron j will result in a large adjustment to its incoming weights. Similarly, large output values, a_j, will result in larger weight adjustments. The learning rate, η, in the weight adjustment equation is selected to reflect the desired convergence speed of the neural network. However, very large values of η lead to instability in the network and unsatisfactory learning while very small values of η give rise to an excessively slow learning rate. Sometimes the learning rate is varied in an attempt to produce a more efficient training technique for the network. For example, allowing the value of η to begin at a high value and to decrease during the learning session can sometimes produce better learning performance [81].

One of the most popular ways to improve the convergence of the weight update is the introduction of a momentum term. The weight adjustment at each iteration is shown in the following equation and is referred to as the 'generalised delta rule' [116]:

$$\Delta w_{ji}^{n} = \eta\, \delta_j a_i + \alpha \Delta w_{ji}^{n-1} \tag{2.3.13}$$

where, α = momentum term constant, $0 \leq \alpha < 1$

Δw_{ji}^{n} = weight update at iteration n

Δw_{ji}^{n-1} = weight update at iteration n-1

Through addition of the momentum term a portion of the weight update is not applied until the following iteration, resulting in dampening of oscillations in weight changes and improved convergence [117].

The weights of the network to be trained are typically initialised at small random values [44]. A well known initialisation technique for a feedforward network with sigmoidal units is to select its weights with uniform probability from an interval ($-\alpha$, α) [58]. A common procedure is to initialise weights to random values between -0.5 and 0.5, or between -1.0 and 1.0, or some other suitable interval [68]. Weight initialisation is an important procedure as it has substantial influence over network convergence. If all weights start out with equal values and if the solution requires that unequal weight values be developed the network may not train properly. Unless random factors, or the random character of input patterns during training, disturb the network, the internal representation may continuously result in symmetric weights. In particular, because the network weight update procedure incorporates the difference between a neuron output and the target output, if each neuron output within the network is the same value, due to equal valued weights, each weight change within the network will be identical and hence, the network weights will never differ. This is not desirable however, as most applications require uneven weights within the network.

Algorithm - The training procedure for the BP neural network is written as follows [72]:

Step 1. Initialise the weights of the network at small random values.

Step 2. Start the learning cycle by exposing the network to a certain input pattern paired with the desired output.

Step 3. Compute the networks output, Equations 2.3.8 and 2.3.9, and compare it with the desired output so that the error can be calculated, Equations 2.3.10 and 2.3.11 for the output and hidden layers, respectively.

Step 4. Adjust the weights of the network using the error BP algorithm so that a certain amount of the detected error is removed, Equation 2.3.12, or 2.3.13 if a momentum term is used.

Step 5. Repeat Steps 2 to 4 with all the input patterns and their corresponding desired outputs (training examples). Compute the cumulative error, Equation 2.3.7.

Step 6. If the cumulative error is within a tolerable range, terminate the training process, otherwise, go back to Step 2.

Applications - BP neural networks have gained significance as an important technique to be applied for various practical applications [55, 118]. BP networks can be applied to almost all applications in the manufacturing domain and in fact, they are the most popular neural network models in manufacturing applications [72]. BP networks are applied to various aspects of manufacturing engineering such as design applications, scheduling, monitoring, diagnosis and quality assurance [119-123].

2.3.3 Radial Basis Function Neural Network

Neural networks based on localised basis functions and iterative function approximation are usually referred to as radial basis function, RBF, networks [76]. A RBF network is a type of feedforward neural network that learns using a supervised training technique. Broomhead and Lowe [124] were the first to exploit the use of RBF's in the design of neural networks. Other major contributions to the theory, design and application of RBF networks include work by Moody and Darkin [125], Renals [126] and Poggio and Gorosi [127]. RBF's are a special class of function, their characteristic feature is that their response decreases monotonically with

increasing distance from a central point [128]. The centre, the distance scale and the precise shape of the radial function are parameters of the model [128]. It has been shown that RBF networks are able to approximate any reasonable continuous function mapping arbitrarily well [125-127, 129] and with the best approximation property [129].

Architecture - A RBF network in its most basic form is comprised of three different layers; an input layer, a hidden layer and an output layer, as shown in Figure 2.3.5. The primary adjustable parameters are the final layer weights, w_{kj}, connecting the k^{th} output node to the j^{th} hidden layer node [130]. There are also weights connecting the input nodes to the hidden layer nodes. All hidden layer nodes are connected to all input nodes and all output nodes. However, there are no connections between non-adjacent layers.

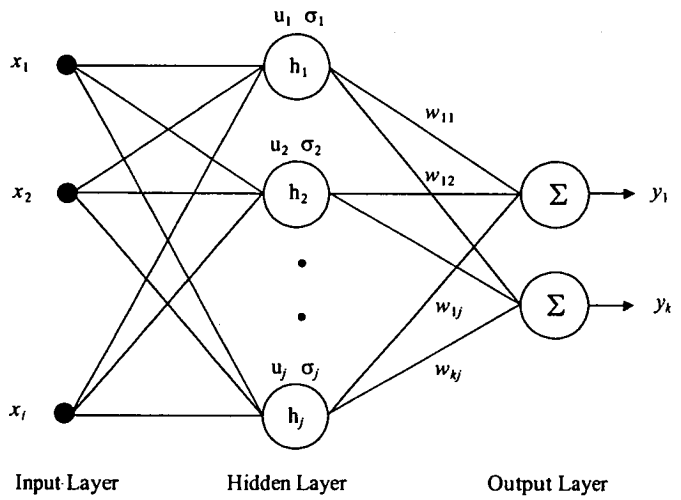


Fig. 2.3.5. Basic Architecture of RBF Neural Network [131]

The function of the input layer is to simply pass forward to all hidden layer neurons the activation patterns applied to the network, hence, it is a linear process. In addition, the purpose of the output layer is to supply the response of the network to the activation patterns applied to the input layer, commonly using a linear function. Generally, a linear weighted summation of the following form is used [119]:

$$y = \sum_{j} h_j w_{kj} \tag{2.3.14}$$

where, h_j = output of hidden layer neuron j, and

w_{kj} = output layer weight

The number of input and output nodes are determined by, and equal to, the number of input and output variables respectively in the process to be modelled. That is, the number of input and output nodes are equal to the dimension of the input and output vectors, respectively. However, the hidden layer neurons perform a non-linear computation and the determination of the number of hidden layer neurons is more complex. Typically, a heuristic approach is required to determine the optimum number of hidden layer nodes in the RBF model. This particular technique is discussed in detail later, as appropriate.

The activation function used in the hidden layer neurons of RBF networks is non-linear and typically, the Gaussian function is used. This function is of an exponential form, as shown in the following equation [71]:

$$h_j = \exp\left[-\frac{(x-u_j)^T(x-u_j)}{2\sigma_j^2}\right] \tag{2.3.15}$$

where, h_j = output of hidden layer neuron j,

x = input vector,

u_j = weight vector of hidden layer neuron j,

T = indicates the vector transpose, and

σ_j = specifies diameter of receptive field of hidden layer neuron j

The shape of the Gaussian function is shown in Figure 2.3.6, for $u = 0.0$ and $\sigma = 1.0$. It can be seen that the function monotonically decreases with distance from the central point.

A RBF network has two distinct operating modes, namely, training and reference. During training, the adjustable parameters of the network, u and σ, and the output layer weight matrix, W, are set so as to minimise the average error between the actual network output and the desired output over the vectors in the training set. Similar to

the BP model, training in a RBF network involves the reduction of prediction error. Consequently, RMS error is again a useful measure of how well the RBF network has learned the training data. In the reference phase, input vectors are applied and output vectors are produced by the network [119].

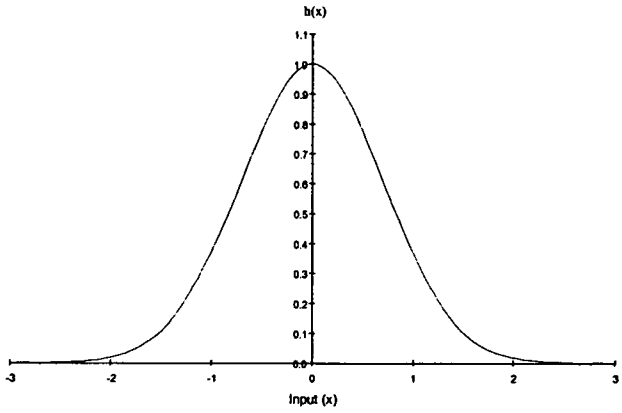


Fig. 2.3.6. Graphical Representation of Gaussian Function using $u = 0.0$ and $\sigma = 1.0$

Training a RBF network involves assigning values for the function centres, u, and the width of the receptive field, σ, for each hidden layer neuron. The location of the centres of the receptive fields is a critical issue and there are many alternatives for their determination [119]. One option is to have a centre and corresponding hidden layer neuron for each input vector in the training set. However, this can lead to a large number of hidden layer neurons and as a result, computation time required is substantially high. An alternative is to locate a hidden layer neuron for a particular cluster of input vectors, significantly reducing the number of hidden layer neurons required and the associated computation time. There are many unsupervised clustering techniques that can be employed to position the centres of the RBF network and adapt the parameters corresponding to the first layer of the network [132]. Alternatively, in an effort to reduce the number of function centres, the easiest and least computational intensive method is that of random selection; selecting m data points randomly from the n training data, $m \leq n$, and using the m data points as function centres [133]. This method assumes that if the n training data patterns form a representative data set, then the randomly selected m data points should also be reasonably representative. However, if the m data points are not representative then the network gives poor approximation as the function centres do not cover the input

range. In choosing the number of basis functions to use, a cross validation technique can be adopted. This involves producing a graph of associated training and test set errors with increasing basis functions, as shown in Figure 2.3.7. When the error becomes constant, that is, the slope of the graph becomes and remains horizontal with increasing basis functions, the minimum number of basis functions to use is the point where the slope of the graph starts to become horizontal. Considering Figure 2.3.7, the optimum number of function centres to use would be approximately 30, as RMS error does not decrease for increasing number of function centres beyond 30.

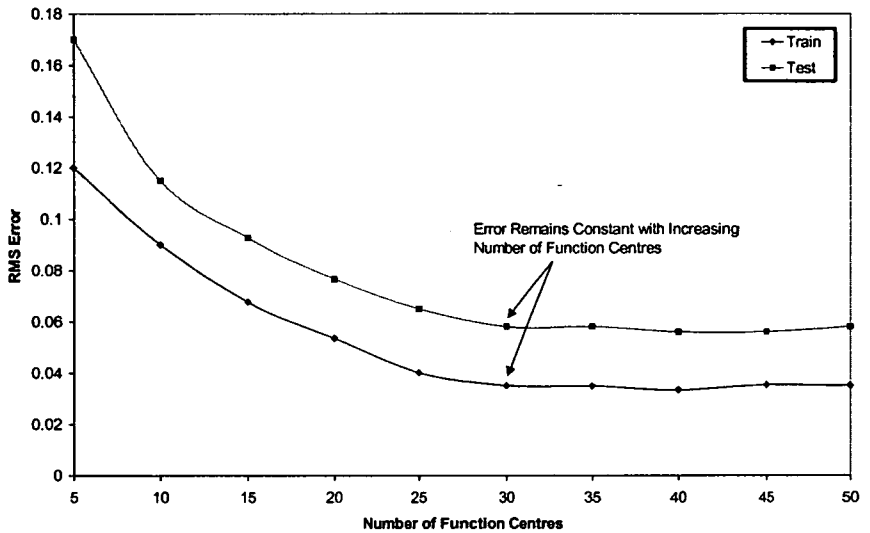


Fig. 2.3.7. Illustrative Graph of RMS Error Behaviour with Increasing Number of Function Centres

On the other hand, using a Kohonen network to cluster the training patterns suitably can be adopted to reduce the required number of function centres. This particular technique is discussed further in a later chapter and a comparison with random selection is presented.

The diameter of the receptive field, σ, can have a profound effect upon the accuracy of the system. For hidden layer neurons whose centres are widely separated from others, σ must be large enough to cover the gap, whereas, those in the centre of a cluster must have a small σ if the shape of the cluster is to be represented accurately [119]. However, there is no formal procedure that exists to establish the optimum σ value to use in the RBF model for a given application, hence, a heuristic approach is

required. Nevertheless, the weight matrix, W, is optimised using a supervised training technique, the generalised delta rule specified for the BP network in the previous section is applicable. Hence, a training set, comprising input vectors and corresponding output vectors, is required for training a RBF network. In RBF network models, the network architecture is determined by the number of nodes in each layer and the location of the function centres [133]. Since the basis function is radially symmetrical, it is usually desirable to normalise the training and test data sets so that each dimension has the same variance [71]. The technique given previously for normalising the data is applicable. This procedure yields a bounded range of 0.0 to 1.0 for the inputs, hence removing any differences in magnitude present among the elements of the input vector.

Algorithm - The training procedure for a RBF neural network is written as follows [72]:

Step 1. Use a suitable clustering technique to set the input-to-hidden layer weights of the network to represent sufficiently the training patterns. Initialise the hidden-to-output layer weights of the network at small random values.

Step 2. Start the learning cycle by exposing the network to a certain input pattern paired with the desired output.

Step 3. Compute the network's output, Equations 2.3.14 and 2.3.15, and compare it with the desired output so that the error can be calculated.

Step 4. Adjust the hidden-to-output layer weights only of the network using the error BP algorithm so that a certain amount of the detected error is removed, Equation 2.3.12, or Equation 2.3.13 if a momentum term is used. The input-to-hidden layer weights remain unchanged from their initial values.

Step 5. Repeat Steps 2 to 4 with all the input patterns and their corresponding desired outputs (training examples). Compute the cumulative error, Equation 2.3.7.

Step 6. If the cumulative error is within a tolerable range, terminate the training process, otherwise, go back to Step 2.

Applications - RBF techniques are powerful methods with a definite range of applicability [119]. The advantage of this particular type of network is in practical application, the basis of its simplicity is that it combines a linear dependence on the variable weights with an ability to model explicitly non-linear relationships [124]. RBF networks have been applied to a wide variety of problems, including; image processing [134, 135], speech recognition [136, 137], time series analysis [138, 139] and medical diagnosis [140].

2.3.4 Kohonen Neural Network

The principal goal of the Kohonen network, also self-organising feature map or topology preserving map, is to transform an incoming signal pattern of arbitrary dimension into a one- or two-dimensional discrete map, and to perform this transformation adaptively in a topological ordered fashion [75]. Teuvo Kohonen [51, 141-144] has been the primary developer of this particular type of network. The Kohonen network uses a sort of competition, called lateral inhibition, to ensure that, when implemented, only the correct neurons in the network become active [145]. Competitive learning is based on the notion that elements in the network must compete among themselves for the privilege of modifying their weights [145]. During training, only the 'winning' neuron and its neighbours within a specified physical distance update the weights on their connections, while the remaining neurons undergo no changes to their connections. The winning neuron at time t is the neuron during training that has a weight vector most closely resembling the input vector given at time t. For each input pattern presented to the network during training, a winning neuron is determined and the corresponding weight vector for that neuron is updated. In addition, for a specified number of neurons in the proximity of the winning neuron, those neurons also undergo a weight update. For a sufficient number of iterations, the network weights will converge to a specific set of values. The number of output neurons, or clusters, for the Kohonen network is specified by the number of categories that the input patterns are to placed into. Hence, subsequent to convergence of the weight matrix, the weight vector for a cluster unit serves as an exemplar of the input patterns associated with that cluster. A significant difference exists between this type of network and the conventional supervised training models in that the correct output cannot be defined *a priori*, hence, a numerical measure of the magnitude of the mapping error is not possible. However, the learning process

leads to the determination of well-defined network parameters for a given application [58].

Architecture - The architecture of the basic Kohonen network consists of only an input layer and an output layer, as shown in Figure 2.3.8. There are i input units and k output units, with all input units being connected to all output units through a weighted connection, w_{ki}.

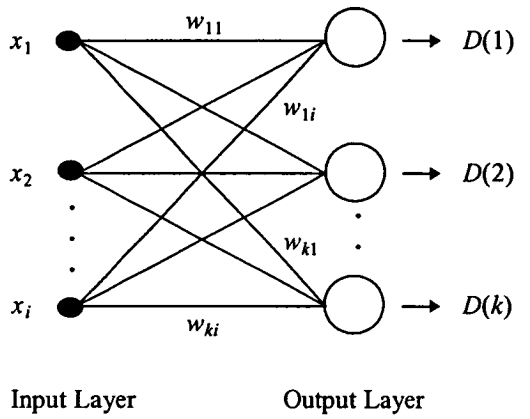


Fig. 2.3.8. Basic Architecture of Kohonen Self-Organising Feature Map

For each real output value from the Kohonen network, $D(1)$ to $D(k)$, there is a minimum value such that the output neuron with the minimum value is the winning neuron. Only that neuron is then allowed to update its associated weights. This is highlighted in the following algorithm.

Algorithm - A training algorithm for the Kohonen neural network is written as follows [68]:

Step 1. Initialise network weights, w_{ki}.
Set topological neighbourhood parameter, R.
Set learning rate parameter, η.

Step 2. While stopping condition is false, do Step 3.

Step 3. For each input vector x, do Steps 4 to 9.

Step 4. For each output neuron k, compute:

$$D(k) = \sum_i (w_{ki} - x_i)^2 \tag{2.3.16}$$

Step 5. Find index k such that $D(k)$ is a minimum.

Step 6. For all output units k within a specified neighbourhood of K, and for all input units i, compute:

$$w_{ki}(new) = w_{ki}(old) + \alpha[x_i - w_{ki}(old)] \tag{2.3.17}$$

Step 7. Update the learning rate and topological neighbourhood parameters.

Step 8. Reduce radius of topological neighbourhood at specified times.

Step 9. Test stopping condition.

The learning process involved in the computation of a feature map is stochastic in nature, which means that the accuracy of the map depends on the number of iterations of the self-organising feature map algorithm [75]. Further, the success of map formation is critically dependant on how the main parameters of the algorithm, namely, the learning-rate parameter, η, and the topological neighbourhood parameter, R, are selected [75]. The learning rate parameter, η, used to update the synaptic weight vector should be time varying [75]. The learning rate parameter should initially start at a value close to unity, then decrease gradually over the period of iterations to a value above 0.1. The exact form of variation of η is not critical, it can be linear, exponential or inversely proportional to the number of iterations [75]. For topological ordering of the weight vectors to take place, careful consideration has to be given to the neighbourhood parameter, R. Initially, the topological neighbourhood parameter should be set to a value such that it includes all neurons in the network, then gradually decreases with increasing iterations such that towards the final iterations the topological neighbourhood parameter includes only the winning neuron and maybe a single neighbour.

Applications - Neural networks developed by Kohonen have been applied to an interesting variety of problems, with one application of this type of network to computer-generated music [68]. Self-organising networks have been implemented successfully for applications including speech recognition [88, 103] and tool wear

pattern recognition for turning operations [146]. Character recognition using the Kohonen self-organising feature map has also had substantial success, using the network to cluster input patterns representing different letters of the alphabet. In addition, the Kohonen network has also been applied successfully to the well-known travelling salesman problem [59].

2.3.5 General Regression Neural Network

The general regression neural network, or GRNN, was discovered by Donald Specht in 1990 [147] and is based on the previously developed Nadaraya-Watson kernel regression [148]. A GRNN is a memory based feedforward neural network, it responds to an input pattern by processing the input data from one layer to the next with no feedback paths. The GRNN is a function approximator system, which is useful for estimating the values of continuous variables such as future position, future values and multi-variable interpolation [149]. GRNN's feature fast training times, can model non-linear functions, and have been shown to perform well in noisy environments given enough data [150]. When using the GRNN, if the variables to be estimated are future values, the GRNN is a predictor. If they are dependent variables related to input variables in a process, plant or system, the GRNN can be used to model the process, plant or system [149]. In both cases the GRNN can instantly adapt to new data points in a very short time by including the new data points in the training set. The primary advantage of the GRNN is the speed at which the network can be trained. Training a GRNN is performed in one pass of the training data through the network; the training data values are copied to become the weight vectors between layers. While the advantages of the GRNN include fast training times, ability to handle both linear and non-linear data and the fact that the smoothing parameter is the only adjustable parameter, thereby making overtraining less likely, the GRNN has some associated disadvantages. For example, the GRNN requires many training samples to adequately span the variation in the data and it requires that all training samples be stored for future use. In addition, the GRNN has trouble with irrelevant inputs and there is no intuitive method for selecting the optimal smoothing parameter.

Architecture - The architecture of a basic GRNN, as shown in Figure 2.3.9, has four layers; input, pattern, summation and output, with weighted connections w_{ji} between

the input and pattern layer and A_i and B_i between the pattern and summation layer. There are i input neurons, j pattern neurons, k+1 summation neurons and k output neurons.

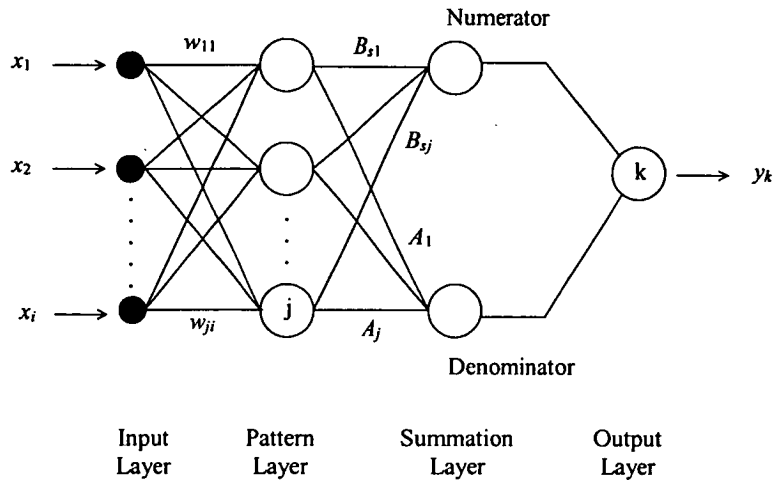


Fig. 2.3.9. Basic Architecture of GRNN

The function of the input layer is to pass forward the activity patterns presented to the network to all neurons in the pattern layer. The number of input layer neurons is equal to the dimension of the input vector. The neurons in the pattern layer perform a non-linear transformation of the input patterns. When a new vector X is entered into the network, it is subtracted from the stored weight vector representing each activity pattern. Either the squares or the absolute values of the differences are summed and fed into a non-linear activation function [147]. The activation function normally used is the exponential function, shown in the following equation:

$$f(net_j) = \exp[\frac{-net_j}{\sigma}] \tag{2.3.18}$$

where, $f(net_j)$ = output from pattern layer neuron j,

net_j = sum of differences between input and weight vector for pattern layer neuron j, and

σ = smoothing factor

The output from all neurons in the pattern layer then become inputs for all neurons in the summation layer. For a single output network the summation layer consists of a denominator and numerator neuron. For each additional output unit a single numerator is added. Hence, the summation layer consists of a single denominator unit and k numerator units, where k equals the number of output neurons. The summation layer neurons perform a dot product between a weight vector and a vector composed of the signals from the pattern units [147]. For the denominator summation neuron, the weight vector is unity, so a simple sum is performed, represented by the following equation:

$$den_{out} = \sum_{j} f(net_j) A_j \tag{2.3.19}$$

where, den_{out} = output from denominator summation neuron,

$f(net_j)$ = output from pattern layer neuron j, and

A_j = weight connecting denominator summation neuron to all pattern layer neurons, equal to 1.0 for all A_j

For the numerator summation neuron, the weight connecting it to each pattern layer neuron is equal to the value of the dependent variable for the training case of that pattern layer neuron [151]. Hence, the numerator summation neuron performs a computation represented by the following equation:

$$num_{outs} = \sum_{j} f(net_j) B_{sj} \tag{2.3.20}$$

where, num_{outs} = output from numerator summation neuron s,

$f(net_j)$ = output from pattern layer neuron j, and

B_{sj} = weight connecting numerator summation neuron s to all pattern layer neurons

The output from the denominator and numerator summation neurons are sent to the output layer neurons, the function of which is to divide the output of the associated

numerator summation neuron by the output of the denominator summation neuron. The result of this division is the output of the GRNN.

Algorithm - As a pre-processing step, it is usually necessary to scale all input variables such that they have approximately the same ranges of variances. The need for this stems from the fact that the underlying probability density function is to be estimated with a kernel that has the same width in each dimension [149]. A training algorithm for the GRNN is written as follows:

Step 1. Determine a suitable value for the smoothing parameter, σ.

Step 2. Set input to pattern layer weights, w_{ji}, equal to the values of the independent variables in the training data set.

Step 3. Set pattern to summation layer denominator weights, A_j, equal to a value of 1.0.

Step 4. Set pattern to summation layer numerator weights, B_{sj}, equal to the values of the output variables in the training data set.

Step 5. Pass the entire training data set through the network and calculate the network output in each instance using Equation 2.3.18 for the pattern layer neurons, and Equations 2.3.19 and 2.3.20 for the summation layer denominator and numerator neurons, respectively.

Step 6. Calculate network output by dividing the numerator output by the denominator output for k output neurons.

Step 7. Compute and observe the prediction error of the network by comparing the predicted output with the target output, Equation 2.3.7.

Step 8. If the prediction error is unacceptable, change the value of the smoothing parameter and repeat Steps 5 to 7.

Step 9. Repeat Step 8 until the prediction error is minimum.

The optimisation of the smoothing factor, σ, is critical to the performance of the GRNN [150]. A useful method for selecting σ is the 'hold out' method, which involves examining the prediction error of the network for different values of σ [149]. The value of σ that produces the lowest error then becomes the selected smoothing factor. This technique is discussed in a later section, as appropriate.

Applications - The GRNN is suitable for prediction, modelling, mapping and interpolation, or as a controller. In particular, the fields of non-linear control systems and robotics are particularly good application areas that can use the potential speed of GRNN's [147]. Narendra and Parthasarathy [152] use a GRNN to separate the problem of control of non-linear dynamical systems into an identification or system modelling section and a model reference adaptive control section. In the identification model, a GRNN was used to approximate the function representing the system behaviour.

2.4 STATISTICAL TECHNIQUES FOR COMPARING POPULATION MEANS

As a part of this literature survey it is necessary to investigate particular statistical techniques that can be used to compare population means. Statistical techniques are required to quantify the statistical significance of the difference between sample means. While the statistical *f*-test [153] and *t*-test [154] are appropriate statistical techniques for comparing two population means, in order to compare several population means simultaneously it is necessary to use a statistical technique called analysis of variance, or ANOVA [155]. ANOVA for comparison of more than two population means is a generalisation of ANOVA for two population means. While ANOVA for two population means is well documented in the published literature it is useful to provide some brief detail here of the generalised ANOVA for comparing means for multiple populations. The purpose of ANOVA is to determine whether the observed difference among sample means is significant. Specifically, ANOVA confirms whether a difference in means is due to random variation or assignable variation. In order to determine this, ANOVA partitions the sum of square of deviations, called the total sum of squares, or *Total SS*, into parts associated with one or more variables plus a remainder that is associated with random error [154]. The total sum of squares is calculated as follows:

$$Total\ SS = \sum_{i=1}^{p} \sum_{j=1}^{ni} \left(x_{ij} - x\right)^2 \tag{2.4.1}$$

where, p = total number of populations,

i = population number, ie. 1, 2, ... , p,

n_i = number in the sample drawn from the i^{th} population,

j = unit in i^{th} population, ie. 1, 2, ... , n_i,

x_{ij} = measured response on the j^{th} unit in the i^{th} sample, and

x = average of all observations

The sum of squares for treatments, *SST*, which is a measure of variation between sample means, is required to determine the test statistic in the ANOVA technique and is calculated using the following formula:

$$SST = \sum_{i=1}^{p} \frac{T_i^2}{n_i} - nx^2 \qquad (2.4.2)$$

where, T_i = mean for observations in the i^{th} population, and

n = total number of observations, ie. $n = n_1 + n_2 + \ldots + n_p$

On the other hand, the sum of squares for error, *SSE*, is a measure of variation within the sample population and is associated with random error. *SSE* is calculated as follows:

$$SSE = \sum_{i=1}^{p} \sum_{j=1}^{ni} (x_{ij} - x)^2 - \sum_{i=1}^{p} \frac{T_i^2}{n_i} - nx^2$$

$$= Total\ SS - SST \qquad (2.4.3)$$

In order to calculate the test statistic for ANOVA it is necessary to determine the mean square for treatments, *MST*, and the mean square for error, *MSE*, which are calculated as follows:

$$MST = \frac{SST}{p-1} \qquad (2.4.4)$$

$$MSE = \frac{SSE}{n_1 + n_2 + \ldots + n_p - p} \qquad (2.4.5)$$

The ANOVA test statistic, F, is the ratio of the mean square for treatments and the mean square for errors, as shown in the following equation:

$$F = \frac{MST}{MSE} \tag{2.4.6}$$

In order to determine the significance of the ANOVA test statistic it is necessary to compare F with a critical value of the F-distribution, F_α. The value of F_α is determined by two degree of freedom values (p-1), represented as ν_1, and (n-p), represented as ν_2, and the confidence level used for the statistical test, α [154]. Typically, a confidence level of 95.0% is used for comparison of population means using ANOVA [156], hence, α equals 0.05. This means that five times out of one hundred a statistically significant difference between means would be found even if there was none, ie. by chance. Further, to interpret the significance of the ANOVA test statistic it is necessary to develop two hypotheses, the null hypothesis and the alternative hypothesis. The null hypothesis, H_o, is that the mean, μ, of each population is equal, ie. $\mu_1 = \mu_2 = \ldots = \mu_n$, while the alternative hypothesis, H_a, is that one or more population means differ. Hence, the two hypotheses describe all possible outcomes of the ANOVA test. The null hypothesis is accepted if F is lower in magnitude than F_α while the alternative hypothesis is accepted if F is higher in magnitude than F_α, as highlighted in the following:

$$\text{null hypothesis} = \begin{cases} H_o, \text{ if } F < F_\alpha \\ H_a, \text{ if } F > F_\alpha \end{cases} \tag{2.4.7}$$

Hence, the result of the statistical test is that all population means are equal or two or more of the population means are statistically significantly different. If the null hypothesis is accepted then it is concluded that there is no statistical significant difference among the population means. However, if the alternative hypothesis is accepted, the ANOVA study does not highlight which population means are statistically significantly different. Consequently, further statistical analysis is required if the alternative hypothesis is accepted as a result of the ANOVA study. In particular, the statistical t-test is one such appropriate methodology that can be used to compare two population means. The t-test assesses whether the means of two

groups of data with unknown means and unknown variances are statistically different from each other [157]. The *t*-statistic is calculated using the following equation [158]:

$$t = \frac{x_1 - x_2}{\sqrt{\frac{S_1^2}{n_1} - \frac{S_2^2}{n_2}}} \tag{2.4.8}$$

where, x_1 and x_2 = mean of population 1 and 2, respectively,

S_1 and S_2 = variance of population 1 and 2, respectively, and

n_1 and n_2 = size of population 1 and 2, respectively

In order to compare all population means it is necessary to complete the *t*-test for each pair of populations. For example, for three populations, p_1, p_2 and p_3, it is necessary to calculate the *t*-statistic for paired populations p_1 and p_2, p_1 and p_3 and p_2 and p_3. However, having completed the *t*-test for p_1 and p_2 to obtain the *t*-statistic, t_{12}, it is not necessary to also determine the *t*-statistic for p_2 and p_1, t_{21}, as it is of equal magnitude, but different in sign, to t_{12}, hence, $t_{12} = -t_{21}$. The *t*-statistic will be positive if the mean of the first population is larger than the second, while negative if it is smaller [159]. Therefore, for p populations requiring pairwise comparison the number of *t*-tests required is $(p)(p\text{-}1)/2$. A *t*-critical value is required to confirm the significance of the *t*-test. This value depends on the significance level, α, used and the degree of freedom, *df*, of the analysis. In order to highlight the importance of selecting an appropriate significance level for the *t*-test it is necessary to introduce the concept of Type I and Type II errors. A Type I error involves incorrectly rejecting a null hypothesis, while a Type II error involves incorrectly accepting a null hypothesis [160]. A Type I error is considered more serious of the two as it draws the conclusion that the null hypothesis is false when, in fact, it is true. Whereas a Type II error is only an error in the sense that an incorrect conclusion was drawn, since no conclusion is drawn when the null hypothesis is not rejected [160]. Hence, it is important that the probability of a Type I error occurring is minimised. Moreover, the probability of a Type I error occurring is given by the significance level, α, and the number of *t*-tests to be completed. In particular, it is calculated using the following relationship:

$$\text{Type I error probability} = 1 - (1 - \alpha)^{(p)(p-1)/2} \tag{2.4.9}$$

It is important to note that while the number of *t*-tests to be completed for pairwise comparison of p populations is fixed at $(p)(p\text{-}1)/2$ the significance level can be carefully selected to minimise the probability of a Type I error occurring. In particular, selecting α equal to 0.01 for the *t*-test considerably reduces the probability of a Type I error occurring. Further, choosing a lower α value actually means that a more rigorous test is performed [161]. Stronger evidence is required to reject the null hypothesis when a significance level of 0.01 is used.

In addition, similar to ANOVA, two hypotheses are required for the *t*-test. The null hypothesis, H_o, is that the mean, μ, of the paired populations are equal, ie. $\mu_1 = \mu_2$, while the alternative hypothesis, H_a, is that the population means differ, ie. $\mu_1 \neq \mu_2$. Hence, the two hypotheses describe all possible outcomes of the *t*-test. The null hypothesis is accepted if the *t*-statistic is lower in magnitude than the *t*-critical value, while the alternative hypothesis is accepted if the *t*-statistic is higher in magnitude than the *t*-critical value, as highlighted in the following:

$$\text{null hypothesis} = \begin{cases} H_o, \text{ if } t-statistic < t-critical \\ H_a, \text{ if } t-statistic > t-critical \end{cases} \tag{2.4.10}$$

Hence, a simple algorithm can be written that details the procedure to follow to compare multiple population means to determine whether a statistically significant difference exists between the means and further, to determine which means are statistically significantly different. This algorithm is written as follows:

Step 1. Complete ANOVA for all populations, p. If $F < F_\alpha$ then the null hypothesis is accepted, ie. no statistical significant difference exists between the population means. However, if $F > F_\alpha$ then the alternative hypothesis is accepted, ie. two or more population means are statistically significantly different.

Step 2. If $F > F_\alpha$ then complete $(p)(p\text{-}1)/2$ *t*-tests in order to determine which population means are statistically significantly different. Confirm the

significance of the t-test by comparing each t-statistic with the t-critical value.

2.5 CONCLUDING REMARKS

While the architecture and algorithms of some particular neural network models are given, it is necessary to investigate the behaviour of these developed models to study their capabilities and limitations. In order to complete this study a sensitivity analysis is useful, using a specified mathematical function where the relationship between the source and response variables is known. This particular numerical investigation is the topic for the following chapter, in which a thorough analysis of each of the specified neural network models is completed. The quantitative analysis of the results incorporates the statistical techniques introduced in this chapter.

CHAPTER THREE

Sensitivity Analysis of Neural Network Models

3.1 INTRODUCTION TO SENSITIVITY ANALYSIS

In order to evaluate the accuracy and limitations of the selected neural network models a mathematical function is formulated and used. Specifically, the sensitivity analysis is used in this instance to assess the suitability of the selected neural networks to model complex, non-linear processes. Hence, this sensitivity analysis is a fundamental study to determine the suitability of the chosen neural networks to model process behaviour at CABBL. In addition, this sensitivity analysis is used to investigate the features of the training data that are used by a particular neural network while making predictions. Generally, when a neural network is initially trained for a particular task, some of the features of the training data will have no significant effect on the networks decision, while other features will be critical. In addition, there exist many networks for a particular task that may perform similarly, however, they may use different features of the training data to make their decision. To illustrate this, consider Figure 3.1.1, which shows a typical feedforward neural network architecture, mapping six inputs, x_1 to x_6, to a single output, Y.

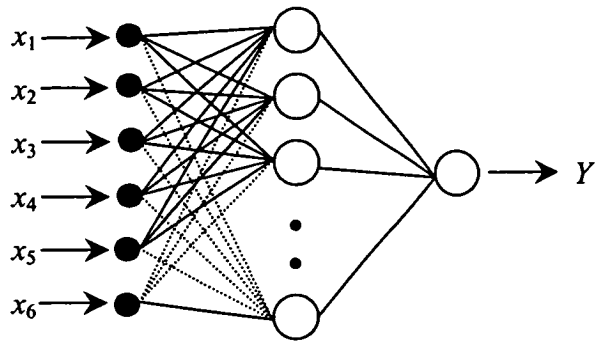


Fig. 3.1.1. Feedforward Neural Network Architecture Showing Six Input Variables, x_1 to x_6, and Single Output Variable, Y

A particular network model, say WH, may only use the input variables x_1 to x_3 to make predictions of Y, ignoring the influence of the remaining variables, x_4 to x_6, for example. When mapping the given function, the WH network may determine that only the features x_1 to x_3 are required to make accurate predictions of Y and hence, identifies x_4 to x_6 as non-contributing inputs. In this instance, x_4 to x_6 could be removed from the data patterns without compromising the accuracy of the network. However, a different network, say RBF, may use the variables x_4 to x_6 to make predictions of Y, ignoring the influence of x_1 to x_3. In this instance, x_1 to x_3 could be removed from the network model without compromising the network accuracy. In either instance, while both the WH and RBF networks may show comparable accuracy, the features of the training data that either network uses to achieve this accuracy may be significantly different. In order to determine the features that are used by a particular neural network model it is important to have some measure of performance. The following section introduces particular sensitivity analysis techniques that can be applied to quantify the degree of influence of input variables on neural network performance.

3.1.1 Introduction to Predictive and Casual Importance Techniques

The two most common measures of input parameter importance are predictive importance and casual importance [162, 163]. Predictive importance is concerned with the increase in generalisation error when a variable is omitted from the training set while casual importance is concerned with situations where the values of the input vector are manipulated in order to investigate the change in the network output. An analysis using predictive importance is completed by initially training the network using all inputs, i. The network is then re-trained with a single input omitted from the model to study the change in the network error, hence, i-1 inputs are used. This is completed i times with a different input omitted in each instance. The resulting change in error in each instance is a direct measure of predictive importance and effect on performance. An increase in error indicates the omitted input is contributing to the networks decision, while a decrease, or no change, indicates the omitted input has no effect on the networks prediction. Alternatively, a relatively simple but effective method for assessing the importance of inputs is the measure of casual importance. Using the casual importance technique, the change in network error as a result of varying individual inputs while other inputs in the training data remain

unchanged is observed. The change in network output corresponding to the given change in the input is directly related to the casual importance of that input. The procedure used to determine the predictive importance and casual importance of input parameters in a neural network model is demonstrated schematically in Figure 3.1.2 to facilitate an understanding of these important techniques.

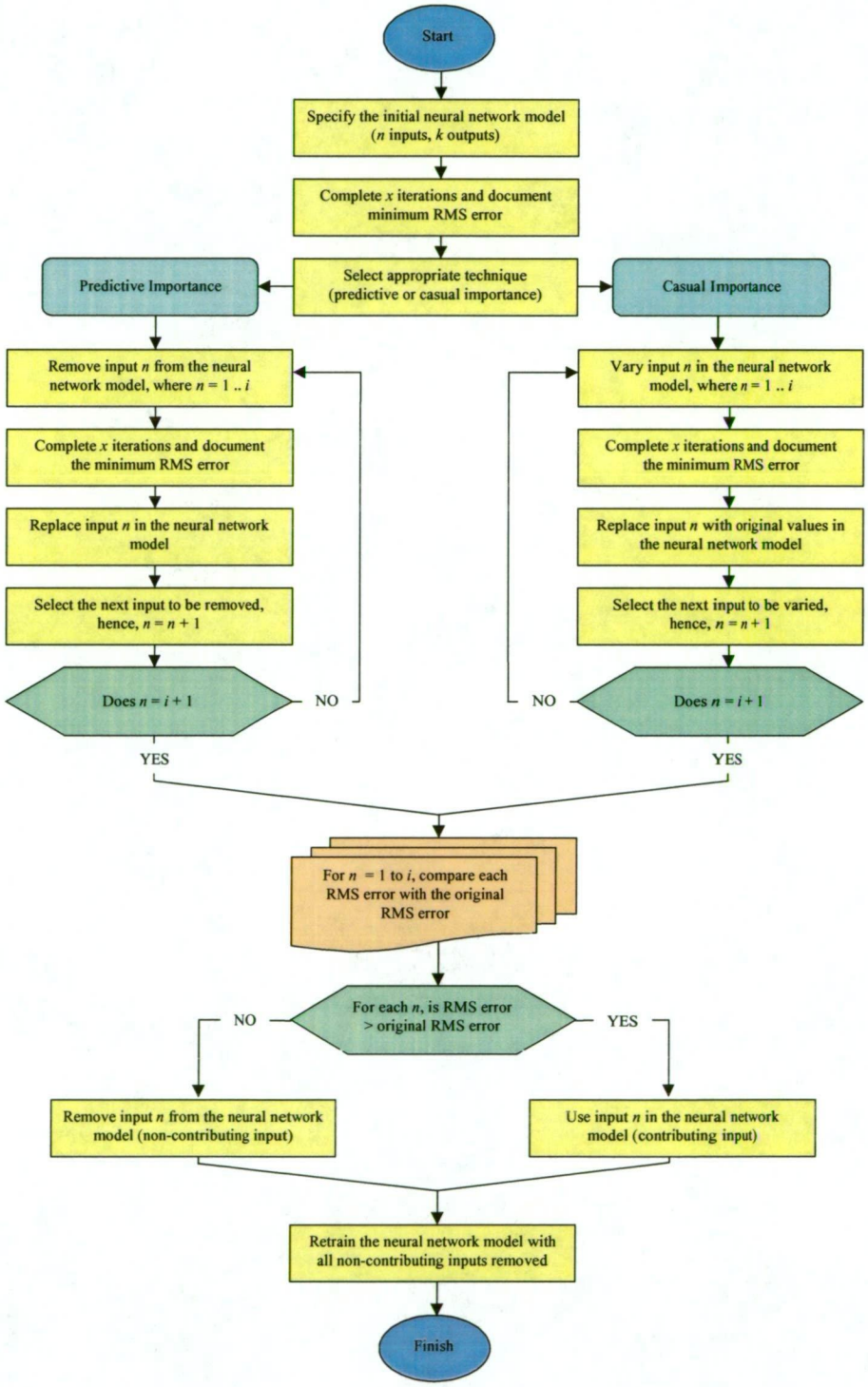


Fig. 3.1.2. Flow Chart Highlighting Algorithm Used to Determine the Predictive and Casual Importance of a Neural Network Input Parameter

It is important to note that predictive and casual importance are different concepts used for different purposes. Predictive importance is the most suitable measure of assessing variable importance in applications where a neural network is used for predictive modelling, trained using observational data where the training cases are a random sample taken from some population [164]. However, in some instances where observational training data is not available, researchers develop values for input variables through scientific experiments. Further, a thorough knowledge of all variables effect on performance has to be determined by means of empirical verification. Predictive importance may not be a suitable measure in this instance as it depends substantially on the experimental design procedure. This particular type of training data produces a casual model, in which the researcher is interested in what happens to the target when one of the input variables is changed. Hence, the measure of casual importance is more applicable. Nevertheless, in a well-designed experiment, the measures of predictive and casual importance will generally produce similar results [164].

The contribution of each input variable to the network prediction can be calculated from the results of the predictive and casual importance analyses. For the predictive importance analysis, this is completed by summing the individual error increase when an input variable is removed from the training and test data sets and calculating the contribution of each input variable to the summed error. Similarly, for the casual importance analysis, this is completed by summing the individual error increase when an input variable is varied in the training and test data sets and calculating the contribution of each input variable to the summed error. Hence, the percentage contribution, *percon*, of each input variable is calculated from the results of the predictive or casual importance analysis using the following formula:

$$percon = \frac{err_{rmsi} \times 100}{\sum_{1}^{i} \left(err_{rmsi} - err_{rms(orig)} \right)} \quad [\%] \qquad (3.1.1)$$

where, i = number of input parameters,

err_{rmsi} = RMS error associated with input i omitted or varied, and

$err_{rms(orig)}$ = original RMS error

However, a condition is required for this equation such that a decrease in RMS error, resulting from the exclusion of a non-contributing input from the data set, does not contribute towards the sum of error increase. The summed value of error increase resulting from omitted inputs must represent only error increases. This is achieved using the following equation:

$$\Delta err_i = \begin{pmatrix} \Delta err_i, \text{if } \Delta err_i \geq 0 \\ 0, \quad \text{if } \Delta err_i < 0 \end{pmatrix} \tag{3.1.2}$$

where, $\Delta err_i = (err_{rmsi} - err_{rms(orig)})$

An omitted input yielding an error decrease, which consequently gives a negative Δerr_i value, has zero percentage contribution in the neural network model. Therefore, using Equation 3.1.2, Δerr_i is set equal to zero for that input and hence, error decreases are not considered in the summed value of error increases.

It is useful to note here that the author has developed and written a specific program for automating the task of data preparation for the predictive and casual importance analyses. Further, the software incorporates an option to automate the calculation of the percentage contribution of the specified input variables. The developed program accompanies this thesis and is named *Neural Network Analysis and Optimisation Strategy* and has the file name *NetAnal.xls*. A detailed user guide for the software is attached in Appendix D. It is also useful to note that the while the accompanying software incorporates further strategies for neural network development and assessment, it is not appropriate to discuss these strategies here. However, it is noted that the particularities and characteristics of these strategies are discussed in a later chapter as appropriate.

3.1.2 Requirements and Specification of Mathematical Function

In order to firstly, comprehensively investigate the sensitivity analysis techniques proposed and secondly, evaluate the ability of the developed neural networks to model complex processes, it is useful to develop a complex mathematical function to produce data sets for neural network training and testing. However, while it has been stated that neural networks are capable of fitting complicated non-linear functions

with correlated inputs, the mathematical function used in this instance has source variables that are statistically independent, in order to simplify this introduction to sensitivity analysis techniques for identifying input variable importance. Hence, the importance of any individual input is not dependent on what other inputs are included in the model or on any interaction between the input parameters. Further, the importance of any one input variable can be assessed without considering the other inputs in the model.

A discontinuous mathematical function is useful in this instance to investigate the ability of the developed neural networks to satisfactorily model a discontinuous function. The discontinuity of a mathematical function is established using the continuity test [188]. The continuity test states that a function, $Y = f(a, b)$, is continuous at $a, b = x$, if, and only if, all three of the following statements are true:

i). $f(x)$ exists (x is in the domain of f)

ii). $\lim_{a,b \to x} f(a,b)$ exists (f has a limit as $a,b \to x$)

iii). $\lim_{a,b \to x} f(a,b) = f(x)$ (the limit equals the function value)

Further, in order to investigate the ability of the neural networks to model non-linear process behaviour it is useful to develop a non-linear mathematical function in this instance. An equation of n variables is defined as linear if it can be expressed in the form [165]:

$$a_1x_1 + a_2x_2 + \ldots + a_nx_n = b \tag{3.1.3}$$

where, $a_1, a_2, \ldots, a_n$ and b are real constants

It is noted that all variables in linear equations occur only to the first power and do not appear as arguments for trigonometric, logarithmic or exponential functions and further, linear equations do not involve any products or roots of variables [165]. On the other hand, equations that do not satisfy these criteria are defined as non-linear.

Hence, in order to satisfy the specified requirements of the mathematical function in this instance, the function developed to complete the sensitivity analysis for the

studied neural networks has five source variables, *a, b, c, d* and *e* and a single response variable, *Y* and is of the form shown in the following equation:

$$Y = \left[\frac{a^5 + b^4 + c^3 + d^2 + e}{((a-1)(b-2)(c-3)(d-4)(e-5))^2} \right]^{0.1} \tag{3.1.4}$$

In the first instance it is noted that the developed mathematical function incorporates variables that are statistically independent. Specifically, the importance of any individual source variable is not dependent on what other source variables are included in the function. Applying a continuity test to the developed function at the points $x = 1, 2, 3, 4$ and 5, yields the following:

a). f is discontinuous at $x = 1$ because $f(1)$ does not exist
b). f is discontinuous at $x = 2$ because $f(2)$ does not exist
c). f is discontinuous at $x = 3$ because $f(3)$ does not exist
d). f is discontinuous at $x = 4$ because $f(4)$ does not exist, and
e). f is discontinuous at $x = 5$ because $f(5)$ does not exist

Attributed to the fact that the function is not continuous at the points $x = 1, 2, 3, 4, 5$, it is shown that the function is indeed discontinuous. A function is only continuous if it is continuous at each and every point of its domain, while the points $x = 1, 2, 3, 4, 5$ are indeed in the domain of the specified mathematical function.

Considering the linearity of the function, it can be seen that the associated variables do not occur only to the first power and further, the mathematical function incorporates products of variables and cannot be expressed in the form noted in Equation 3.1.3. Hence, the function does not satisfy the criteria for linearity and is therefore non-linear. It is useful to note that while the mathematical function used in this instance is developed to satisfy the criteria of statistical independence, discontinuity and non-linearity, there are many other equally valid mathematical functions that could have been used. However, the developed mathematical function specified in Equation 3.1.4 is a suitable example in this instance.

3.1.3 Neural Network Training and Test Procedure

Numerical data for the specified mathematical function was generated using a random number generator for the source variables, subsequently calculating the value of the response variable according to the function given in Equation 3.1.4. However, it is useful to note that the random number generator was bounded to the range 0.0 to 50.0 for each of the source variables. It is important to note that a random number generator is used in this instance to generate numerical values for the mathematical function as no previous data exists, as the function is chosen arbitrarily to satisfy the specified criteria rather than being an actual model of some process. Further, the objective of bounding the values of the source variables is to simulate an actual process, as typically, process parameters are bound between some minimum and maximum values of an operating range. Hence, while the purpose of the mathematical function is to provide a means of studying the effect of changes in the source variables on the output parameter, the mathematical function with randomly generated values represents, quantitatively, real values that may be encountered in an actual process. It is useful to note that 1,200 data patterns were developed in total for this sensitivity analysis, divided into 1,000 training patterns and 200 test patterns. While the training data set is used to teach the mathematical function to the selected neural network models, the test data set is used to assess network performance. It is important to note that the test data is an entirely different data set within the boundaries of the training data and is therefore an independent validation set for assessing network performance. Figure 3.1.3 shows the range of output values used in the training and test data sets and highlights the uniform distribution of data in either instance. The train and test data patterns were filtered from a larger data set such that a uniform distribution of the output variable was obtained for the train and test data sets. Many researchers [48, 44, 63, 68] highlight the importance of using a uniform distribution of the data for neural network training and testing rather than a normal distribution.

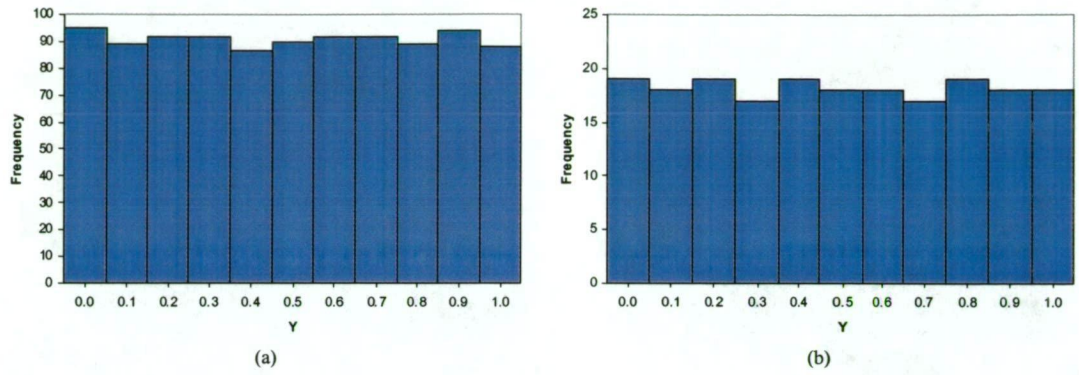


Fig. 3.1.3. Histogram Showing Distribution of the Network Output Variable in the Data Sets. (a) Training, and (b) Test

In preparing the data for presentation to the network models, the input and output, or target, data were normalised to achieve unit variance. In this instance, all network data was scaled to the bounded range 0.0 to 1.0. Further, normalisation of the output data used for training the network is necessary as most training algorithms minimise the total error of all outputs. If the output variables are unequally scaled, those with larger variability will be favoured, as they will dominate the error sum [18, 207]. Hence, in this instance the output variable was also normalised in the bounded range 0.0 to 1.0 in the data sets.

As a further measure of network performance, two additional source variables, *f* and *g*, were added to the training and test data. However, the value of the response variable was not altered for the addition of these two inputs. While a random number generator was used to create values for *f* and *g*, the value of *Y* remained unchanged from the value calculated using only the five initial source variables, *a, b, c, d* and *e*. The objective of adding these variables is to investigate the ability of a neural network to ignore the influence of non-contributing inputs while making predictions and hence, determine the underlying relationship given by the original specified mathematical function.

While the measures of predictive and casual importance can be used to determine the ranking of importance of the input variables, it is useful to consider the ranking of the source variables of the specified mathematical function using a traditional statistical technique, specifically, a multi-variable regression analysis. While the ranking of

importance of the source variables can be determined in this instance through observation of the specified mathematical function, statistical techniques, such as regression analysis, can also be applied to determine this order. In addition, the ranking of importance of the input variables using this technique will serve as a useful comparison for the results of the predictive and casual importance techniques applied later in the chapter.

3.1.4 Multi-Variable Regression Analysis

A statistical analysis is used to determine the effect of the source variables on the response variable for the specified mathematical function. However, it is important to note that the statistical technique applied in this is used merely to compliment the neural network sensitivity analysis methodology discussed. The superiority of neural networks, compared to traditional statistical techniques, to accurately model the complex specified mathematical function is highlighted in a later section. The particular statistical test used in this instance is a multi-variable regression analysis, MVRA. This particular test is used to determine the effect that each of the source variables, *a, b, c, d, e, f* and *g*, in the specified mathematical function have on the single response variable, *Y*. The multi-variable regression equation is of the following form [166]:

$$Y' = a + b_1X_1 + b_2X_2 + \ldots + b_kX_k \tag{3.1.5}$$

where, Y' = predicted value for variable Y

a = intercept constant

k = number of independent variables

b_1 to b_k = regression coefficients for the k independent variables, and

X_1 to X_k = values of the k independent variables

It can be seen from the regression equation that there is a regression coefficient associated with each of the source variables. These coefficients highlight the degree of dependence of the performance feature on the independent variables. These coefficients are the weights associated with the corresponding source variable, when used in combination with the other specified source variables. It is important to note that if source variables are either added or removed from the equation then the values

of the regression coefficients will change. In addition to the regression coefficients, the output of the MVRA produces a correlation coefficient, *r*, which is an index that summarises the magnitude of the relationship between a response variable and the specified source variables, considered simultaneously [167]. The correlation coefficient is a number between -1.0 and 1.0 that measures the association between paired values. The closer the coefficient is to -1.0 or 1.0 the stronger the association between the variables and the closer the scatter of points is to a straight-line [168]. The statistical significance of each regression coefficient is evaluated using a statistical *t*-test, which checks the unique contribution of each source variable, indicating whether each regression coefficient is significantly different from zero. The null hypothesis for this test is that the coefficient is not statistically significantly different from zero, while the alternative hypothesis is that the coefficient is statistically significantly different from zero. A *t*-critical value of 2.326 is determined using a '*critical values of t*' table [166] for a significance level of 0.01 and sample size of 1,000, using the training data set. Therefore, the magnitude of the *t*-statistic is required to be higher than 2.326 for a regression coefficient to be significantly different to zero. The results of the MVRA completed for the specified mathematical function are shown in Table 3.1.1.

TABLE 3.1.1. Multi-Variable Regression Analysis Results for Specified Mathematical Function

Regression Statistics			
r: 0.7908	r^2: 0.6254	*adjusted* r^2: 0.6232	*observations*: 1,000
Source Variable	Regression Coefficient	*t*-statistic	Significant
a	0.00132	7.70972	yes
b	-0.00299	-17.38061	yes
c	-0.00390	-22.54617	yes
d	-0.00424	-24.92863	yes
e	-0.00439	-25.91982	yes
f	0.00005	-0.28453	no
g	0.00002	-0.12520	no

It is shown from the value of the *adjusted* r^2 term, 0.6232, that the level of accuracy of the MVRA is not particularly high. Nevertheless, it is shown as a result of the MVRA that the source variables *f* and *g* have no significance in the specified mathematical function, confirmed by the *t* statistic value being lower than the *t*-critical value in either instance. However, the remaining source variables are shown

to have significance in the mathematical function and can be ranked from highest to lowest significance, using the calculated regression coefficients. Calculating the sum of the absolute values of the regression coefficients then dividing the regression coefficient for each source variable by the sum allows a comparison of the magnitude of the regression coefficients using a percentage contribution of each source variable to the sum. It was found that the most significant source variable is *e* (26.0%), followed by *d* (25.2%), then *c* (23.2%) and then *b* (17.7%), with *a* (7.8%) being least significant of the contributing source variables. In addition, *f* and *g* were found to have no significance in the specified mathematical function, both yielding a percentage contribution of 0.0%. It is useful to note that the ranking of the input variables using the MVRA is in agreement with the ranking observed through visual inspection of the specified mathematical function. It can be seen from visual inspection that variation of *a* will have little effect on the dependent variable while the most significant changes in *Y* result from variation of *e*. The results obtained from this MVRA are useful later for a comparison with the results obtained from the predictive and casual importance techniques when utilised for neural network modelling.

In addition to observing the ranking of importance of the source variables using this technique, it is also interesting to investigate the accuracy of the MVRA for predicting values of the response variable, given only the source variable values. In order to compare the accuracy of the MVRA with that of the developed neural network models used in this investigation, it is useful to consider the RMS error associated with the regression analysis. The MVRA for the specified mathematical function has shown a RMS error value of 0.0871 for the training data set and 0.0936 for the test data set. The RMS error values shown here represent the error between the actual specified value of the response variable, *Y*, calculated using the mathematical function, and the predicted values of *Y*, using the MVRA, based on the values of the source variables. The RMS error obtained using the MVRA is useful to compare with the RMS error obtained using the developed neural network models to compare the accuracy of these paradigms. While both techniques have the same objective, the modelling characteristics of either are significantly different.

3.1.5 Experimental Procedure for Sensitivity Analysis

The sensitivity analysis in this instance studied the capabilities of six different neural network models. It is useful to note that as a part of this work the application of these neural networks to the mathematical function to investigate the influence of input variables on a performance feature has been published in international refereed conference proceedings [163, 169] where the associated methodology and issues have been debated and accepted by prominent artificial intelligence researchers. While the particular neural networks used for this sensitivity analysis are those listed in the following, it is important to note that the rationale for using these specific models has been provided in Chapter Two. Nevertheless, the neural networks applied for the sensitivity analysis in this instance include:

- Widrow-Hoff (WH) Neural Network
- Backpropagation - 1 hidden layer - (BP1) Neural Network
- Backpropagation - 2 hidden layers - (BP2) Neural Network
- Radial Basis Function (RBF) Neural Network
- Radial Basis Function - incorporating Kohonen - (RBFKOH) Neural Network
- General Regression Neural Network (GRNN)

While each model is a particular type of feedforward network incorporating supervised training, the architecture and algorithms of the various models are distinctly different. In addition, the architecture of the various networks was modified in each instance to investigate the resulting effect on network performance. Further, for the RBF model the effect of clustering the inputs, to obtain the network weights, compared to selecting randomly from the training patterns, was investigated. It has been noted in a previous section that selecting training patterns as network weights randomly from the training data can lead to poor approximation if the randomly selected patterns are not representative of the entire training data. It has been stated that a preferable method is to cluster similar training patterns as a means of improving the performance of the RBF model. As a means of clustering the training patterns, a Kohonen neural network is utilised. In addition, the effect of using different activation functions in the various layers of the network models is also studied. In particular, the WH, BP1 and BP2 network models are compared for a logistic and linear activation function in the output layer of the network. However, all

of the network models, regardless of structural differences, perform the same task in this instance. The objective of the individual networks is to model the specified mathematical function and minimise the error between the target values and the network predicted values. Particular aspects of neural computation being assessed in this sensitivity analysis include:

i). ability of each network to accurately model the specified mathematical function

ii). ability of each network to learn the underlying relationship, ignoring the effect of the non-contributing variables, *f* and *g*, in the specified mathematical function

iii). determine the features of the input data set that each network uses for predictions of the output variable, and

iv). computation time required for the individual network models to learn the specified mathematical function.

Moreover, it is useful to note that the sensitivity analysis is completed for all the applied neural network models using the following procedure:

Step 1. Present the specified mathematical function, using the input variables *a, b, c, d, e, f* and *g* and output variable *Y*, to each of the network models. For each model, determine the most appropriate network architecture, based on a study of RMS error.

Step 2. Using the network architecture that produces minimum RMS error, for each network model, use the predictive importance technique to determine the contribution of each input variable to network predictions. Subsequently, remove any non-contributing variables from each of the applied networks and re-train the models using the remaining input variables to obtain a corresponding RMS error value for the train and test data sets.

Step 3. Complete a sensitivity analysis, using the casual importance technique, to quantitatively determine the degree of importance of each contributing input variable in the network model.

The capability of each of the applied neural networks is indicated using a quantitative and qualitative measure. The qualitative measure indicates how closely the network can identify the shape of the specified mathematical function while the quantitative measure indicates the prediction error of the network. A quantitative and qualitative comparison of each applied neural network is given in a later section of this chapter.

3.2 SENSITIVITY ANALYSIS RESULTS AND DISCUSSION

For each of the neural network models applied, the results are presented in the following format. Initially, the RMS error behaviour is shown for the various architectures used for each neural network. In each instance, the network architecture varies by the number of nodes used in a particular layer of the network and in some instances by the output layer activation function used. From the range of network architectures used, an architecture producing minimum RMS error is chosen as the best architecture for each network model. That particular architecture is then used to complete an analysis of input variable importance, using the predictive and casual importance techniques. Further, an important consideration in selecting a particular network model for an application is the computation time required. Computation time is an important consideration in neural network modelling as it specifies the time required by a network to converge to a suitable model of a particular function or process. While computation time is not such an important consideration for applications where the neural network is trained 'off-line' and used as a static model, high computation times can be a substantial disadvantage in instances where the network model is required to train and be used as a dynamic model 'on-line', for real time prediction or classification applications. It is important to note that the measure of computation time as a part of this investigation is for comparative purposes rather than conclusive. While it is understood that computation time is highly dependent on the processing speed of the particular computer used to execute the neural network software program, it is noted that the computation time required for each of the developed neural network models was assessed using the same computer processing unit. Hence, the computation time measured for each model can be assessed to give a direct comparison of time required by each network to model the specified mathematical function. It is also important to note that computation time in this instance is the time required for the network parameters to converge to an optimum model of the mathematical function.

3.2.1 Widrow-Hoff Neural Network (WH)

The architecture of the WH network is shown in Figure 3.2.1 to highlight that this particular network has an input and output layer only, no hidden layer is required. Hence, there is no decision to make on the number of hidden layer nodes required as a means of optimising the network accuracy. Further, the number of input and output layer nodes required in the WH model are determined by the number of input and output variables, respectively, associated with the application. Specifically, the number of input layer nodes required is equal to 7 in this instance, as there are 7 input variables, while there is only 1 output layer node required, as there is a single output parameter.

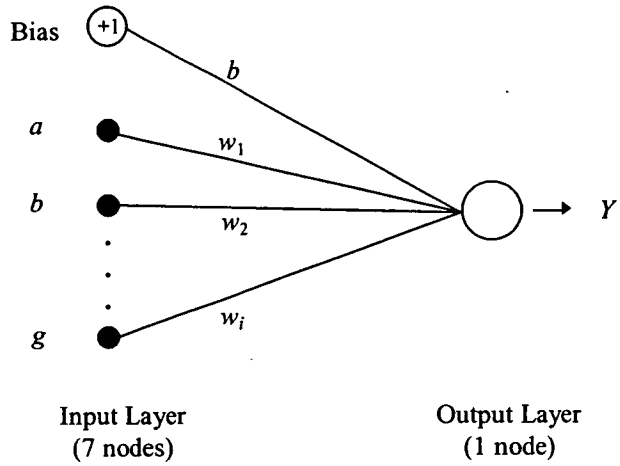


Fig. 3.2.1. Architecture of WH Neural Network used for Sensitivity Analysis

However, in order to optimise the network accuracy it is useful to compare the RMS error of the model for a sigmoidal, or logistic, activation function and a linear summation activation function in the output layer. The specific values of RMS error and computation time obtained using the different activation functions are shown in Appendix B, Table B.1. It can be seen that a lower RMS error was obtained for the sigmoidal activation function in the output layer for the train and test data sets, 0.0843 and 0.0909, respectively, compared to 0.0968 and 0.1032 for the train and test data sets, respectively, for the linear activation function. It is important to note that the learning rate, α, used in the delta rule for the weight update procedure is an adjustable parameter, generally set to a value in the range 0.1 to 0.9 [63]. The convergence speed of the neural network is directly related to the value of α. In particular, it has been noted that if α is small then the search path will closely

approximate the gradient path, but convergence will be very slow due to the large number of update steps required to reach a local minima. On the other hand, if α is large then convergence will initially be very fast, but the algorithm will eventually oscillate and thus not reach a minimum [170]. However, the modified neural network in this instance incorporates a procedure to decrease a large initial value of α to a small value, not less that 0.1, as network iterations progress. This technique allows large weight changes to occur when the search point is far away from the minimum, decreasing to a smaller value as iterations progress and the search approaches the minimum, optimising network convergence speed and consequently, minimising computation time. Hence, the initial value of α is set to 0.9, decreasing to 0.1 as network iterations progress. While the particular algorithm used to decrease α is documented in the WH program code, it is useful to note here that a linear technique is adopted.

The behaviour of RMS error with increasing iterations is shown in Figure 3.2.2 for the WH model.

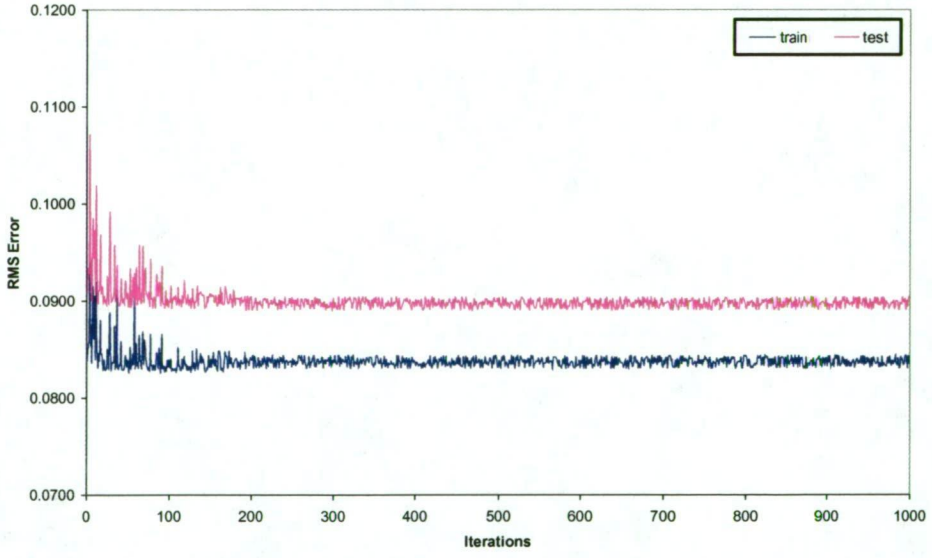


Fig. 3.2.2. WH Network RMS Error Behaviour with Increasing Iterations (7 input layer nodes, sigmoidal activation function in output layer)

It can be seen that RMS error decreases rapidly initially and achieves a minimum value for the train and test data sets after approximately 200 iterations, beyond which RMS error is shown to remain constant, reflecting convergence of the neural network

connection weights. Hence, approximately 200 iterations ensure convergence of the network weights in this instance, with no requirement to increase the number of training iterations beyond this number. Hence, computation time is documented for 200 iterations for this particular network model.

As the sigmoidal activation function in the output layer produced minimum RMS error then this particular network architecture is used for an analysis of input variable importance using the predictive importance technique. Further, a statistical analysis of the error resulting from the individual removal of the input parameters confirms the statistical significance of the observed increase or decrease in error. In particular, a statical *t*-test is used in this instance to compare the population means from the original and modified data sets for the removal of each input. Hence, the error associated with the neural network model when no input parameters are omitted is compared with the error associated with each model corresponding to an omitted input parameter. It is important to note that the test data set error is used for this statistical analysis and is calculated using the following equation:

$$err_i = pred_i - targ_i \tag{3.2.1}$$

where, err_i = error associated with test data pattern i,
$pred_i$ = predicted value for test data pattern i, and
$targ_i$ = actual value of test data pattern i

The significance of the *t*-statistic is evaluated using a *t*-critical value to determine whether the null hypothesis, which states that the population means are equal, or the alternative hypothesis, which states that the population means are statistically significantly different, is accepted. The null hypothesis is accepted if the *t*-statistic is lower in magnitude than the *t*-critical value, while the alternative hypothesis is accepted if the *t*-statistic is greater in magnitude than the *t*-critical value. For a significance level of 0.01 and a degree of freedom, *df*, of 199 a *t*-critical value of 2.345 is appropriate. Table 3.2.1 shows the results from the predictive importance analysis, highlighting the train and test RMS error obtained when each of the 7 input variables are alternatively omitted from the train and test data sets and the

corresponding *t*-statistic value, the percentage contribution of each input parameter and associated computation time.

TABLE 3.2.1. WH Network Predictive Importance Results (*t*-critical = 2.345, α = 0.01, *df* = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0843	0.0909	-	-	129.24
a	0.0875	0.0928	4.360	2.0	119.90
b	0.0955	0.1034	7.319	13.2	119.86
c	0.1025	0.1120	17.265	22.3	119.02
d	0.1056	0.1179	21.508	28.5	118.26
e	0.1071	0.1230	32.184	33.9	117.70
f	0.0841	0.0907	1.627	0.0	119.25
g	0.0841	0.0907	1.618	0.0	118.19
non-cont. variables omitted	0.0839	0.0905	2.512	-	109.26

It can be seen that the removal of *a* from the train and test data sets produced a slight increase in RMS error, 0.0875 and 0.0928 for the train and test sets, respectively, while the removal of *e* produced the most significant change in RMS error, 0.1071 and 0.1230 for the train and test data sets, respectively. In addition, individually omitting *b*, *c* and *d* from the data sets also produced an increase in RMS error in each instance. Further, the statistical analysis has shown that the increase in error associated with individually omitting the input parameters *a*, *b*, *c*, *d* and *e* is statistically significant as the alternative hypothesis is accepted in each instance. However, the individual removal of *f* and *g* from the data sets result in no change in RMS error, confirmed by the *t*-statistic value being lower in magnitude than the *t*-critical value in each instance. It is shown that omitting *f* from the data sets produced a train and test RMS error of 0.0841 and 0.0907, respectively, while omitting *g* produced the same result. Hence, this result indicates that *f* and *g* are not contributing towards the network prediction accuracy, therefore, *f* and *g* can be permanently removed from the training and test data patterns. Thus, the network is trained and tested with *f* and *g* collectively removed from the data patterns, producing an improved RMS error of 0.0839 and 0.0905 for the train and test data sets, respectively. The *t*-statistic value confirms that this decrease in error is statistically significant.

In order to complete an analysis of casual importance, the contributing variables, *a*, *b*, *c*, *d* and *e*, in the data sets are individually varied across the minimum to maximum values in their respective ranges while all other variables remain at their original values in each instance. The associated RMS error and *t*-statistic value for each varied input parameter are shown in Table 3.2.2. It can be seen that varying *a* produces a slight increase in RMS error, 0.0980 and 0.1040 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.1132 and 0.1271 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when varied. The statistical significance of the observed increase in error is confirmed by the alternative hypothesis being accepted in each instance.

TABLE 3.2.2. WH Network Casual Importance Results (*t*-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Varied in Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0980	0.1040	5.129	10.2	111.67
b	0.1011	0.1094	8.155	14.3	108.56
c	0.1096	0.1186	17.858	21.3	109.83
d	0.1129	0.1254	22.617	26.4	110.32
e	0.1132	0.1271	32.972	27.7	109.81

From the percentage contribution results it can be seen that the most important variable influencing the WH network prediction is *e*, followed by *d*, *c*, *b* and *a*. In addition, the variables *f* and *g* are shown to have no influence on network predictions. It is useful to note that the results from the predictive and casual importance techniques in this instance are comparable to the ranking of importance of the input variables obtained using the MVRA and from visual inspection of the specified mathematical function.

A study of computation time behaviour with changing network architecture has shown computation time to decrease with decreasing number of input layer nodes, as shown in Table 3.2.1. For instance, using the sigmoidal activation function in the output layer and 7 variables in the input data sets, computation time is shown to be 129.24 seconds, decreasing to an average of 118.88 seconds for 6 variables in the input data set and further decreasing to 109.26 seconds for 5 variables in the input

data set. Hence, while it is shown that the removal of non-contributing variables from the training and test data is desirable for improving network accuracy, it is also useful to note that it is beneficial for decreasing computation time. In addition, it is shown that computation time is less for the linear activation function in the output layer, 74.54 seconds for 7 input variables, compared to the sigmoidal activation function. As the linear activation function is not as computationally intensive as the sigmoidal function then it follows that computation time is lower when the linear activation function is used.

3.2.2 Backpropagation - 1 Hidden Layer - Neural Network (BP1)

In order to facilitate a discussion of the features of the BP1 neural network that were adjusted in this instance to determine the minimum RMS error for the model, it is useful to consider the network architecture used for this sensitivity analysis, as shown in Figure 3.2.3.

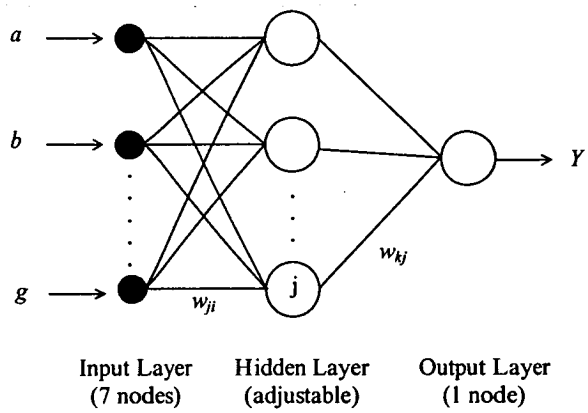


Fig. 3.2.3. Architecture of BP1 Neural Network used for Sensitivity Analysis

It can be seen that optimising the architecture of the BP1 network is slightly more complex compared to the WH model. In particular, for the BP1 network it is necessary to vary the number of nodes in the hidden layer in order to determine the most suitable architecture for producing minimum RMS error. A technique that is appropriate to achieve this is to plot RMS error for an increasing number of hidden layer nodes. The graph is analysed to determine the architecture producing minimum error. This technique is completed in this instance for the BP1 network for the sigmoidal and linear activation function alternatively in the output layer, while maintaining a sigmoidal activation function in the hidden layer nodes, as shown in

Figure 3.2.4. In addition, RMS error and computation time obtained for the BP1 model using an increasing number of hidden layer nodes for the two output layer activation functions used are shown in Appendix B, Table B.2. It is shown that 7 hidden layer nodes produced the lowest RMS error, using a sigmoidal activation function in the output layer, giving a RMS error of 0.0447 and 0.0636 for the train and test data sets, respectively. Hence, this particular architecture is used for the measure of input importance in the following section.

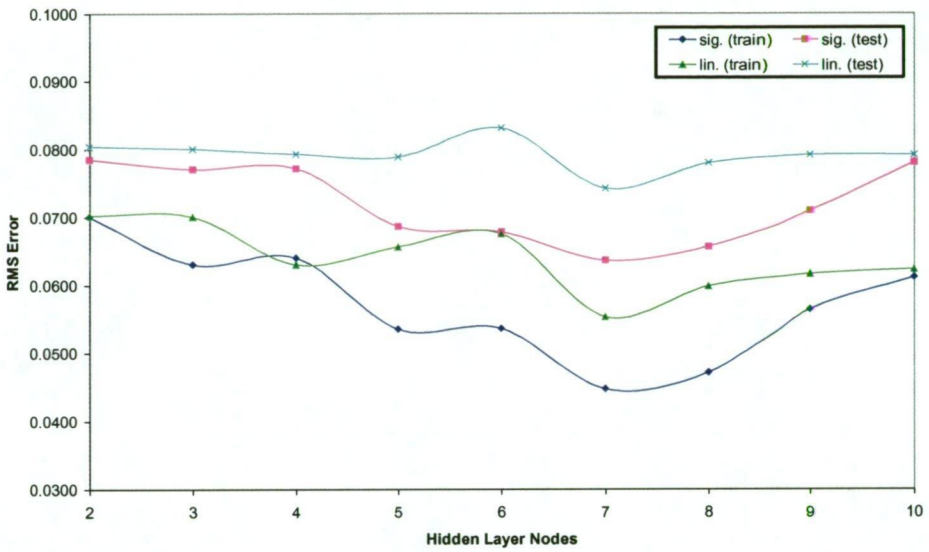


Fig. 3.2.4. BP1 Network RMS Error Behaviour with Changing Architecture

It is important to note that, similar to the WH network, the learning rate, α, used in the delta rule for the weight update procedure was decreased during network training from a large initial value of 0.9 to a minimum value of 0.1 as training iterations increased. RMS error behaviour with an increasing number of iterations has shown that approximately 600 iterations are required to allow convergence of the network weights, producing a minimum RMS error for the network predictions. As shown in Figure 3.2.5, RMS error decreases rapidly during the initial 100 iterations. In addition, RMS error is shown to remain uniform after approximately 600 iterations are complete, not decreasing further as iterations increase to 1,000. Hence, computation time is noted for 600 iterations for this particular network model.

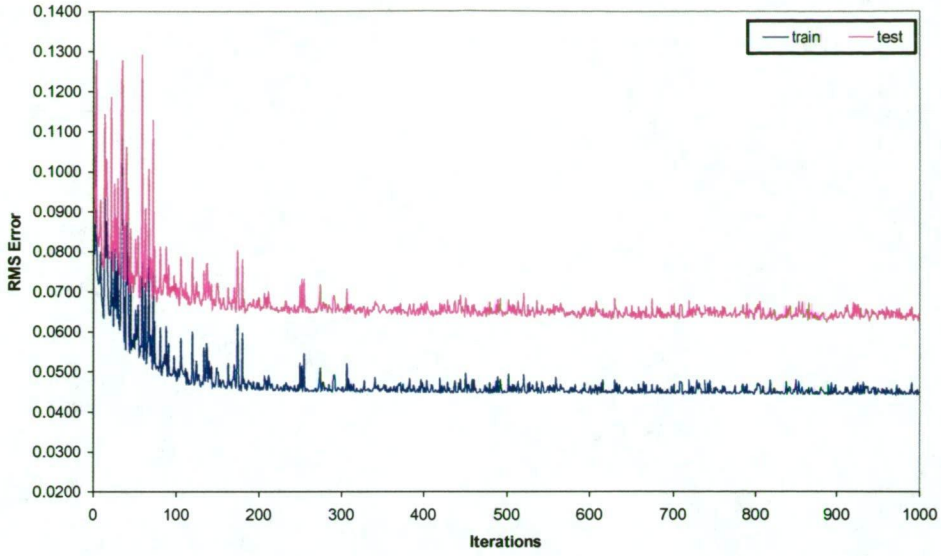


Fig. 3.2.5. BP1 Network RMS Error Behaviour with Increasing Iterations (7 input layer nodes, 7 hidden layer nodes, sigmoidal activation function in hidden and output layers)

Similar to the WH network, it is shown in Table 3.2.3 that the removal of *a* from the data sets produced a slight increase in RMS error, 0.0623 and 0.0766 for the train and test data sets, respectively. On the other hand, omitting *e* produced the most significant increase in RMS error, 0.0899 and 0.1081 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when omitted from the data sets. While a *t*-critical value of 2.345 is appropriate, it is shown that the *t*-statistic value is higher in magnitude in each instance, confirming the increase in error to be statistically significant. In addition, the removal of both *f* and *g* from the data sets yields a decrease in RMS error in each instance, also shown to be statistically significant. It is shown that the error decreases to 0.0447 and 0.0628 for the train and test data sets, respectively, for the removal of *f* and 0.0446 and 0.0629 for the train and test data sets, respectively, for the removal of *g*. Hence, *f* and *g* are collectively removed from the train and test data sets in order to complete the sensitivity analysis using the measure of casual importance. The resulting RMS error from omitting *f* and *g* from the data sets is shown to statistically significantly decrease to a value of 0.0442 and 0.0629 for the train and test data sets, respectively. The contribution of each input variable on the network prediction, using the predictive importance technique, is shown in Table 3.2.3.

TABLE 3.2.3. BP1 Network Predictive Importance Results (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0447	0.0636	-	-	943.29
a	0.0623	0.0766	6.325	7.8	864.07
b	0.0768	0.0933	8.588	17.9	865.31
c	0.0835	0.0983	13.074	20.9	864.06
d	0.0839	0.1078	19.146	26.6	863.95
e	0.0899	0.1081	29.410	26.8	865.42
f	0.0447	0.0628	2.507	0.0	865.19
g	0.0446	0.0629	2.782	0.0	864.96
non-cont. variables omitted	0.0442	0.0629	3.104	-	822.72

The casual importance measure of the input variables, *a*, *b*, *c*, *d* and *e*, as shown in Table 3.2.4, has shown that varying *a* produces a slight increase in RMS error, 0.0633 and 0.0768 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.0910 and 0.1138 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when varied. The alternative hypothesis is accepted for each t-test, confirming the statistical significance of the RMS error increase in each instance. The contribution of each input variable on the network prediction, using the measure of casual importance, is shown in Table 3.2.4.

TABLE 3.2.4. BP1 Network Casual Importance Results (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0633	0.0768	7.672	7.2	825.63
b	0.0825	0.0973	9.341	17.8	821.49
c	0.0882	0.1060	11.228	22.3	818.76
d	0.0910	0.1136	19.989	26.3	822.45
e	0.0910	0.1138	31.380	26.4	824.16

It can be seen from the percentage contribution results that the most important variable influencing the BP1 network prediction is *e*, followed by *d*, *c*, and *b*, while *a* is shown to be least significant of the contributing variables.

A study of computation time behaviour with changing network architecture has shown that computation time increased with increasing number of hidden layer

nodes. For a sigmoidal activation function in the output layer and 7 variables in the input data sets, computation time is initially 315.33 seconds for 2 hidden layer nodes, increasing to 1,286.38 seconds for 10 hidden layer nodes. In addition, it is shown that while computation time is lower for the linear activation function in the output layer compared to the sigmoidal activation function, the same trend of increasing computation time with increasing number of hidden layer nodes is observed for the linear activation function. The results have shown computation time to decrease with decreasing number of input layer nodes, again highlighting the benefit of only using the minimum number of input variables required to model a particular function, achieved by omitting non-contributing variables from the network.

3.2.3 Backpropagation - 2 Hidden Layers - Neural Network (BP2)

The BP2 neural network used in this instance is shown in Figure 3.2.6, highlighting the particular architecture used for this sensitivity analysis.

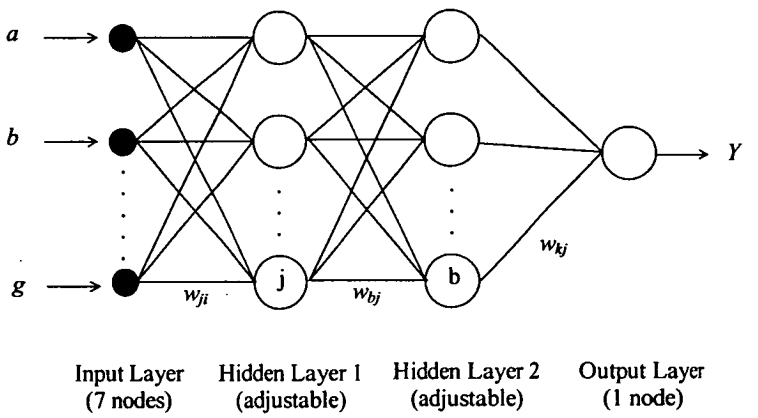
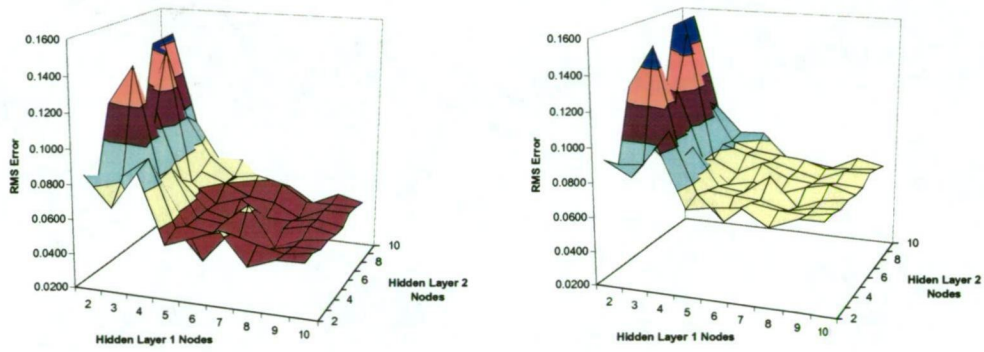


Fig. 3.2.6. Architecture of BP2 Neural Network used for Sensitivity Analysis

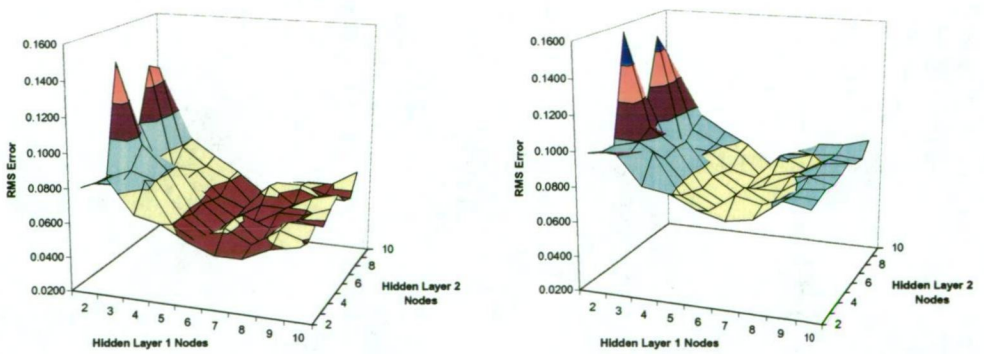
It can be seen that in addition to the complexities involved with the selection of an optimum network architecture described for the single hidden layer BP1 network, the BP2 network involves the optimisation of an additional network layer. Similar to the technique discussed for selecting the optimum number of hidden layer nodes in the BP1 network for producing minimum RMS error, the RMS error is plotted against increasing number of hidden layer nodes for the BP2 model until no change or an increase in RMS error is observed. With the inclusion of the additional hidden layer, the graph becomes three-dimensional. However, the technique remains appropriate for this particular network and is completed in this instance for a sigmoidal and linear

activation function in the output layer, Figures 3.2.7 and 3.2.8, respectively. In addition, Appendix B, Tables B.3.1 and B.3.2 show the RMS error and computation time obtained using the sigmoidal and linear activation function in the output layer, respectively. It is shown that the best architecture for achieving minimum RMS error is using a sigmoidal activation function in the output layer, with 6 nodes in the first hidden layer, followed by 3 nodes in the second hidden layer. The resulting RMS error using this particular architecture is shown to be 0.0388 and 0.0603 for the train and test data sets, respectively.



(a) (b)

Fig. 3.2.7. BP2 Network RMS Error Behaviour with Changing Architecture for Sigmoidal Output Layer Activation Function. (a) Training, and (b) Test



(a) (b)

Fig. 3.2.8. BP2 Network RMS Error Behaviour with Changing Architecture for Linear Summation Output Layer Activation Function. (a) Training, and (b) Test

For this particular neural network model, the learning rate, α, was again decreased from 0.9 to 0.1 as the network training iterations progressed. Figure 3.2.9 shows the behaviour of RMS error with an increasing number of iterations over the range 0 to 1,000 iterations. It can be seen that approximately 900 iterations are required to allow convergence of the network weights. Hence, computation time is noted for this particular neural network model for 900 iterations.

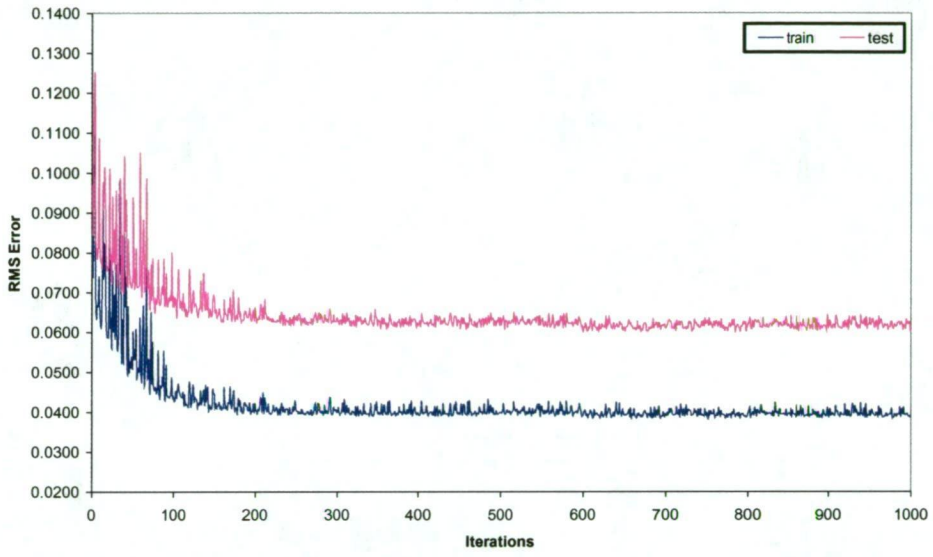


Fig. 3.2.9. BP2 Network RMS Error Behaviour with Increasing Iterations (7 input layer nodes, 6 first hidden layer nodes, 3 second hidden layer nodes, sigmoidal activation function in hidden and output layers)

Completing an analysis of input importance using the predictive importance technique, as shown in Table 3.2.5, it can be seen that the removal of *a* from the data sets produced a slight increase in RMS error, 0.0619 and 0.0859 for the train and test data sets, respectively. In addition, omitting *e* produced the most significant increase in RMS error, 0.0872 and 0.1115 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when omitted from the data sets. For a *t*-critical value of 2.345 the null hypothesis is rejected for each population, hence, the observed error increase in each instance is confirmed to be statistically significant. Further, the *t*-test has shown that the removal of both *f* and *g* from the data sets yielded no change in RMS error in either instance. Hence, the network is re-trained with *f* and *g* collectively removed from the data sets,

producing a statistically significantly improved RMS error of 0.0375 and 0.0597 for the train and test data sets, respectively.

TABLE 3.2.5. BP2 Network Predictive Importance Results (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0388	0.0603	-	-	1459.62
a	0.0619	0.0859	6.318	12.9	1324.27
b	0.0737	0.0892	9.925	14.5	1322.89
c	0.0842	0.1050	13.004	22.5	1326.82
d	0.0843	0.1089	18.609	24.4	1319.64
e	0.0872	0.1115	24.738	25.7	1323.90
f	0.0385	0.0602	0.742	0.0	1321.08
g	0.0384	0.0603	0.805	0.0	1324.72
non-cont. variables omitted	0.0375	0.0597	2.497	-	1182.46

A casual importance measure of the input variables, as shown in Table 3.2.6, has shown that varying *a* produces a slight increase in RMS error, 0.0582 and 0.0824 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.0858 and 0.1104 for the train and test data sets, respectively. Further, the results highlight that the variables *b*, *c* and *d* produce an increase in RMS error when varied. Further, the alternative hypothesis is accepted for each t-test, confirming the error increase to be statistically significant in each instance.

TABLE 3.2.6. BP2 Network Casual Importance Results (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0582	0.0824	6.513	11.7	1184.08
b	0.0756	0.0899	9.228	15.6	1181.50
c	0.0854	0.1027	14.081	22.2	1182.91
d	0.0795	0.1066	21.547	24.2	1182.47
e	0.0858	0.1104	24.738	26.2	1183.63

From the percentage contribution results it is shown that the most important variable influencing predictions of *Y* is the input variable *e*, followed by *d*, *c*, then *b* with *a* being least significant of the contributing variables. It is also shown that the variables

f and *g* have no influence on the network predictions. Further, it is shown that the measures of predictive and casual importance are in close agreement with the previously studied models regarding the ranking of importance of the input variables.

Computation time behaviour with increasing number of hidden layer nodes has shown comparable results to the previous studied models. Specifically, computation time is shown to increase with increasing number of hidden layer nodes. For a sigmoidal activation function in the output layer of the network and 7 variables in the input data sets, computation time is shown to be initially 563.54 seconds for 2 nodes in the first and second hidden layers of the network, increasing to 3,342.86 seconds when 10 nodes are used in each hidden layer of the network. Likewise, the results have shown computation time to decrease with decreasing number of input layer nodes and also, computation time is lower when a linear activation function is used in the output layer of the network, compared to the sigmoidal activation function. It is highlighted from these results that computation time, as expected, is proportional to the complexity of the network architecture. Decreasing the number of nodes in a network decreases network complexity and hence, decreases the computation time associated with training a network, as does decreasing the complexity of the activation function used in the processing nodes.

3.2.4 Radial Basis Function Neural Network (RBF)

The architecture of the RBF network used for this sensitivity analysis is shown in Figure 3.2.10. While the activation function used in the hidden layer of the RBF network was the Gaussian function, the output layer activation function used consistently in this work was the linear summation function. However, the network architecture was varied by the number of hidden layer nodes used and the width of the receptive field, σ, of the Gaussian function. While the weights connecting the input and hidden layer nodes are fixed, the values of the weights were set using patterns selected randomly from the training data in this instance. The minimum number of hidden layer nodes required and the optimum value for σ were selected by plotting the behaviour of RMS error for an increasing number of hidden layer nodes and increasing σ, over the range 0.1 to 0.9, as shown in Figure 3.2.11. Appendix B, Table B.4 shows the specific values of RMS error and computation time associated with changes in the network architecture. It is shown that the RMS error is minimum

for 50 hidden layer nodes and σ equal to 0.6. Hence, this particular architecture is used for an analysis of input importance using the predictive and casual importance techniques. It can be seen that the train and test RMS error values produced using this architecture are 0.0683 and 0.0833, respectively.

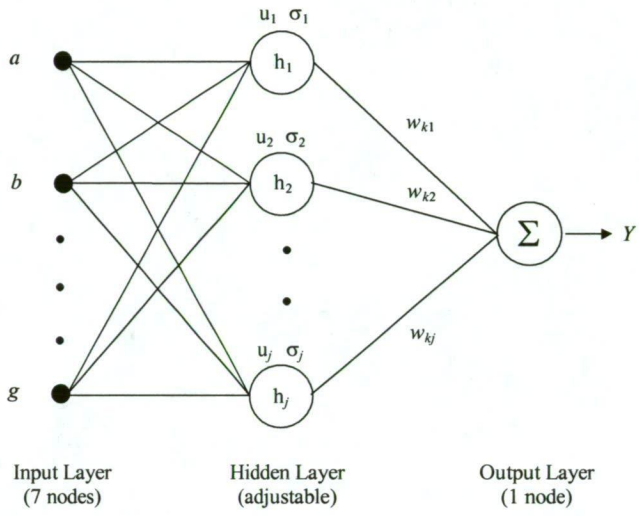
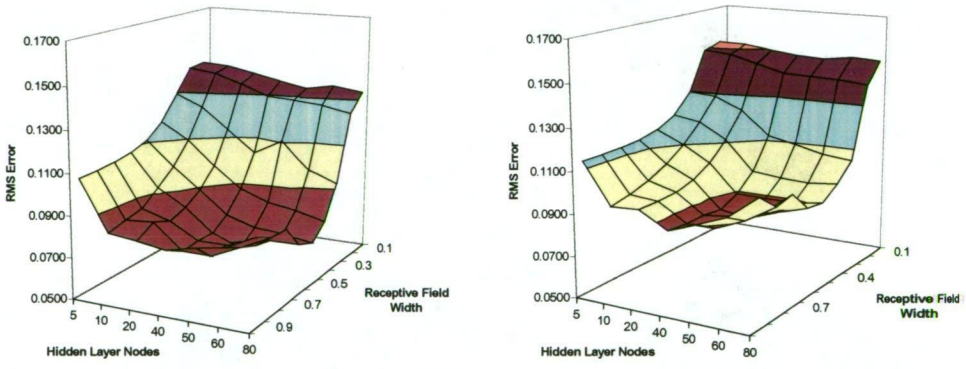


Fig. 3.2.10. Architecture of RBF Neural Network used for Sensitivity Analysis



(a) (b)

Fig. 3.2.11. RBF Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

It is important to note that while the weights connecting the input layer to the hidden layer are fixed, the hidden to output layer weights are adjusted during network training. The adjustable parameter for the weight update procedure is the learning rate, α, which was initially 0.9, decreasing to 0.1 as training iterations progressed. A plot of changing RMS error with an increasing number of iterations, as shown in Figure 3.2.12, has shown that RMS error decreases rapidly during the initial 50

iterations and remains uniform after approximately 200 iterations. Hence, 200 iterations are used to determine the required computation time for this particular network.

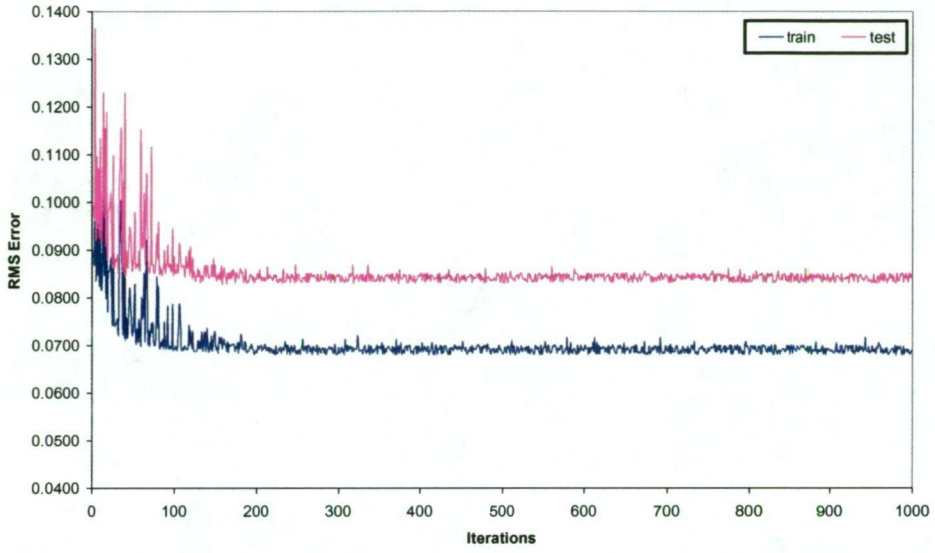


Fig. 3.2.12. RBF Network RMS Error Behaviour with Increasing Iterations (7 input layer nodes, 50 hidden layer nodes, σ equal to 0.6, Gaussian and linear activation function in hidden and output layer, respectively)

Prior to documenting the results of the importance analysis it is interesting to note that the value of σ has a significant influence on the accuracy of the network prediction. This is highlighted in the change in error for the range of σ values used. For small σ, equal to 0.1, the associated RMS error is shown to be 0.1324 and 0.1476 for the train and test data sets, respectively, for 50 hidden layer nodes. However, the RMS error is shown to decrease to 0.0683 and 0.0833 for the train and test data sets, respectively, when σ is increased to 0.6. For σ equal to 0.9 the RMS error is shown to increase to 0.0814 and 0.0924 for the train and test data sets, respectively. This behaviour highlights the importance of selecting an appropriate value for σ for accurate neural network modelling. RMS error variation with changing σ is attributed to the fact that σ controls the shape of the Gaussian function. By changing σ the shape of the Gaussian function is changing as a consequence. For a particular application a specific Gaussian function shape will produce a minimum error for the network model, which is determined by selecting an appropriate value of σ.

The measure of predictive importance using the specified architecture is completed to highlight the ordering of importance of the 7 input variables. It is shown in Table 3.2.7 that an increase in RMS error for the train and test data sets, 0.0905 and 0.1017, respectively, is associated with the removal of *a* from the data sets, while the most significant increase in RMS error is associated with the removal of *e* from the data sets, 0.1078 and 0.1302 for the train and test data sets, respectively. It is also shown that omitting *b*, *c* and *d* from the data sets produces an increase in RMS error, in ascending order. In addition, it is shown that the removal of *f* and *g* from the data sets produces a decrease in RMS error, 0.0658 and 0.0827 for the train and test data sets, respectively, for the removal of *f* and 0.0654 and 0.0825 for the train and test data sets, respectively for the removal of *g*. The *t*-statistic has shown that the error change associated with the individual removal of each input is statistically significant. Hence, the input variables *f* and *g* are removed from the data sets to produce an improved RMS error of 0.0634 and 0.0807 for the train and test data sets, respectively, also shown to be statistically significant. The contribution of each input variable to the network prediction, using predictive importance as a measure, is highlighted in Table 3.2.7.

TABLE 3.2.7. RBF Network Predictive Importance Results (*t*-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0683	0.0833	-	-	1084.60
a	0.0905	0.1017	8.755	10.5	1042.43
b	0.0975	0.1125	18.129	16.7	1041.27
c	0.1040	0.1197	24.330	20.8	1042.58
d	0.1064	0.1272	27.439	25.1	1042.90
e	0.1078	0.1302	31.548	26.8	1041.94
f	0.0658	0.0827	3.054	0.0	1042.08
g	0.0654	0.0825	2.982	0.0	1041.86
non-cont. variables omitted	0.0634	0.0807	3.673	-	992.84

The casual importance measure of the input variables, as shown in Table 3.2.8, highlights that varying *a* produces an increase in RMS error, 0.0994 and 0.1117 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.1197 and 0.1412 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* produced an increase in

RMS error when varied. The *t*-statistic is higher in magnitude than the *t*-critical value in each instance, confirming the error increase associated with varying each input variable to be statistically significant.

TABLE 3.2.8. RBF Network Casual Importance Results (*t*-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Varied in Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0994	0.1117	6.108	12.7	990.74
b	0.1085	0.1225	15.592	17.1	991.51
c	0.1172	0.1343	19.837	21.9	992.37
d	0.1180	0.1380	24.374	23.5	990.68
e	0.1197	0.1412	28.620	24.8	991.46

The results have highlighted that the most important variable influencing network predictions is *e*, while *a* is least significant of the contributing variables. Further *d*, *c* and *b* are shown to contribute in descending order of importance. In addition, it is shown that the variables *f* and *g* have no influence on network predictions of *Y*. The ordering of importance of the input variables obtained using the RBF network is in close agreement with the WH, BP1 and BP2 neural network models and the MVRA.

Similar to the WH, BP1 and BP2 models, a study of computation time behaviour with changing network architecture has shown computation time to decrease with decreasing number of input layer nodes. In addition, while the value of the receptive field width is shown to have no influence on computation time, it is shown that computation increases with increasing number of hidden layer nodes. For 7 variables in the input train and test data sets and σ equal to 0.6, computation time is shown to be 115.80 seconds for 5 hidden layer nodes, increasing to 1,660.42 seconds for 80 hidden layer nodes. This confirms the need to reduce the complexity of the network architecture to a minimum without compromising network accuracy.

3.2.5 Radial Basis Function - incorporating Kohonen - Neural Network (RBFKOH)

Similar to the RBF network discussed in the previous section, the activation function used in the hidden layer of the RBFKOH network was the Gaussian function, while the output layer activation function used was consistently the linear summation

function. Likewise, the network architecture was varied by the number of hidden layer nodes used and the value of σ in the Gaussian function. The particular network architecture used for this model is the same as that shown in Figure 3.2.10. However, in this instance, while the weights connecting the input and hidden layer nodes are fixed, the values of the weights were determined using a Kohonen neural network to cluster similar training patterns, using the clustered patterns as the weight values. While the previous network used patterns selected randomly from the training data as network weights, it is useful to use clustered patterns as network weights to identify any improvement in performance that may result.

The learning rate, α, used in updating the hidden to output layer weights was decreased from 0.9 initially to 0.1 as training iterations progressed. The number of hidden layer nodes required and the value of σ were determined by plotting the behaviour of RMS error for an increasing number of hidden layer nodes and increasing σ, over the range 0.1 to 0.9, as shown in Figure 3.2.13. Appendix B, Table B.5 also shows the RMS error and computation time for the changing network architecture.

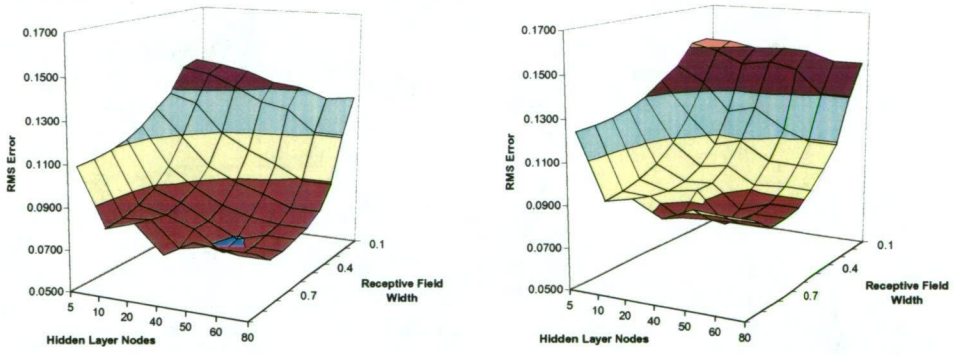


Fig. 3.2.13. RBFKOH Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

The architecture producing minimum RMS error is found to be 50 hidden layer nodes, with a σ value of 0.7. It is shown that the RMS error associated with this particular architecture is 0.0665 and 0.0798 for the train and test data sets, respectively. A study of RMS error behaviour with changing σ has shown error to vary significantly with changing σ. It is shown that RMS error decreases rapidly in

each instance from a maximum value associated with a σ value of 0.1 to a minimum value for σ equal to approximately 0.7. Further, it is shown that RMS error increases slightly for σ values greater than 0.7. Again this highlights the need to select carefully the value of σ used in the neural network model.

RMS error behaviour with an increasing number of iterations, as shown in Figure 3.2.14, has shown RMS error to decrease rapidly during the initial 100 iterations, remaining uniform after approximately 200 iterations are complete. It is shown, using 1,000 iterations in total, that RMS error does not decrease beyond 200 iterations. Hence, computation time is noted for 200 iterations for this particular neural network model.

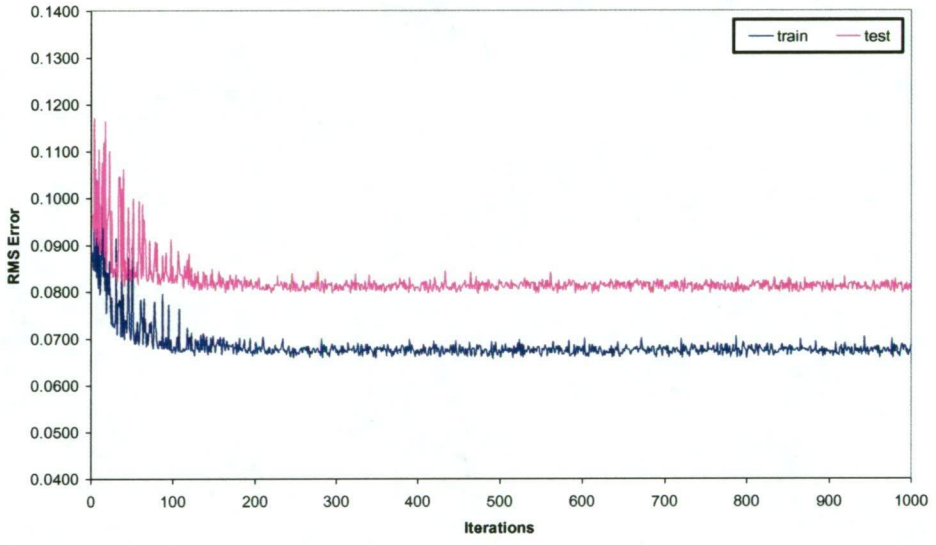


Fig. 3.2.14. RBFKOH Network RMS Error Behaviour with Increasing Iterations (7 input layer nodes, 50 hidden layer nodes, σ equal to 0.7, Gaussian and linear activation function in hidden and output layer, respectively)

An analysis of input importance was completed using the predictive importance technique for the particular network architecture described, as shown in Table 3.2.9. It can be seen that the removal of *a*, *b*, *c*, *d* and *e* from the data sets produced an increase in RMS error. The initial RMS error of 0.0665 and 0.0798 for the train and test data sets, respectively increased to 0.0789 and 0.0891 for the train and test data sets, respectively, corresponding to the removal of *a*. Omitting *e* produced the most significant increase in RMS error, resulting in values of 0.1137 and 0.1393 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each

produced an increase in RMS error when omitted from the data sets. The increase in error resulting from the removal of *a*, *b*, *c*, *d* and *e* from the data sets is shown to be statistically significant, as the alternative hypothesis is accepted for each *t*-test completed for these input parameters. On the other hand, the individual removal of *f* and *g* from the data sets has shown no statistically significant change in RMS error in each instance. However, re-training the network with both *f* and *g* removed from the data sets has shown an improved RMS error of 0.0624 and 0.0789 for the train and test data sets, respectively, which is shown to be statistically significant. The contribution of each input variable on network predictions is shown in Table 3.2.9 for the predictive importance technique.

TABLE 3.2.9. RBFKOH Network Predictive Importance Results (*t*-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0665	0.0798	-	-	3090.51
a	0.0789	0.0891	5.093	5.3	2695.11
b	0.0917	0.1066	7.265	15.2	2695.49
c	0.0985	0.1145	11.840	19.7	2695.87
d	0.1077	0.1259	16.542	26.1	2695.16
e	0.1137	0.1393	18.914	33.7	2695.18
f	0.0655	0.0797	0.641	0.0	2696.05
g	0.0653	0.0794	1.237	0.0	2695.64
non-cont. variables omitted	0.0624	0.0789	2.618	-	2645.60

The casual importance measure of the input variables, as shown in Table 3.2.10, highlights that varying *a* produces the least significant increase in RMS error, 0.0860 and 0.0973 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.1275 and 0.1455 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when varied. The statistical analysis has confirmed the observed increase in error in each instance to be statistically significant. It is shown from the percentage contribution of the input parameters that the most important variable influencing the prediction of *Y* using the RBFKOH model is *e*, followed by *d*, *c* and *b*, while *a* is shown to be least significant of the contributing variables. In addition, it is shown that the variables *f* and *g* have no influence on network

predictions. These findings are comparable to the results obtained using the WH, BP1, BP2 and RBF neural networks and the MVRA.

TABLE 3.2.10. RBFKOH Network Casual Importance Results (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0860	0.0973	5.774	8.6	2645.78
b	0.0927	0.1059	8.312	12.6	2645.85
c	0.1067	0.1248	12.563	21.4	2645.07
d	0.1138	0.1351	15.085	26.2	2645.23
e	0.1275	0.1455	19.223	31.1	2645.72

A study of computation time for the RBFKOH model has shown computation time is significantly increased for this model compared to the RBF network, which can be attributed to the use of the Kohonen network for clustering the training data. It is shown for 7 variables in the input data sets and σ equal to 0.7 that computation time is 370.34 seconds for 5 hidden layer nodes, increasing to 4,693.45 seconds for 80 hidden layer nodes. In addition, it has been shown that computation time decreases with decreasing number of input layer nodes. The major implication of this finding is that while RMS error improves as a result of using the Kohonen clustering technique, computation time is significantly increased as a consequence.

3.2.6 General Regression Neural Network (GRNN)

It can be seen from Figure 3.2.15 that the GRNN model used for this sensitivity analysis is a four layer network. However, as discussed in the previous chapter, the only layer in the network that can be varied to optimise network error is the pattern layer. However, the value of σ used in the activation function in this layer can be varied to produce minimum RMS error. In this instance, σ controls the shape of the exponential function. RMS error is plotted for an increasing number of pattern layer nodes and increasing σ value over the range 0.1 to 0.9, as shown in Figure 3.2.16. The RMS error and computation time associated with the changing network architecture are also shown in Appendix B, Table B.6. It can be seen that a minimum RMS error of 0.0483 and 0.0925 for the train and test data sets, respectively, is achieved using 600 pattern layer nodes and a σ value of 0.2. Similar to the RBF network models studied, the GRNN model has shown large changes in RMS error for

changing σ. It is shown that RMS error is high for large σ values, decreasing as the value of σ decreases, to a minimum error corresponding to a σ value of 0.2. This behaviour confirms the importance of correctly selecting the σ value if high accuracy is to be achieved in the neural network models.

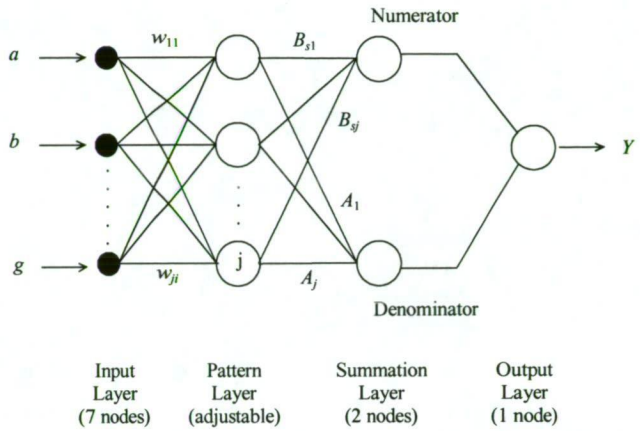
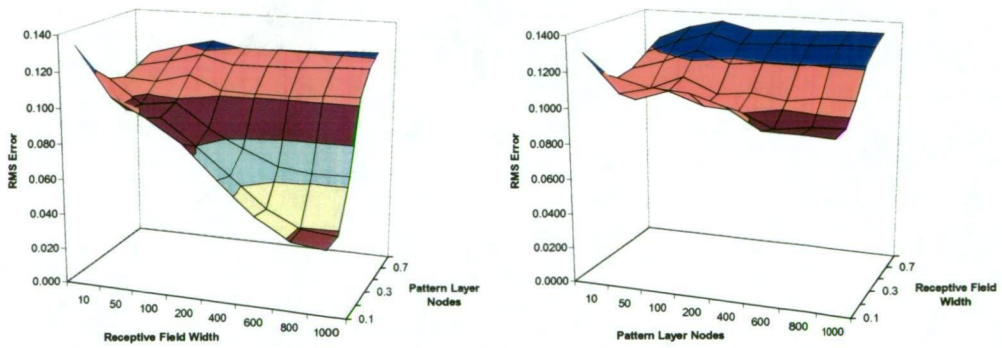


Fig. 3.2.15. Architecture of GRNN Network used for Sensitivity Analysis



(a) (b)

Fig. 3.2.16. GRNN Model RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

A predictive importance analysis using the specified network architecture, as shown in Table 3.2.11, highlights that an increase in RMS error for the train and test data sets, 0.0555 and 0.0942, respectively, is associated with the removal of input variable *a*. On the other hand, the most significant increase in RMS error is associated with the removal of *e*, 0.0658 and 0.1164 for the train and test data sets, respectively. It is also shown that individually omitting *b*, *c* and *d*, from the data sets produces an increase in RMS error. Conversely, it is shown that the individual removal of *f* and *g*

from the data sets produces a decrease in RMS error in each instance. The error is shown to decrease to 0.0481 and 0.0891 for the train and test data sets, respectively, for the removal of *f* and 0.0483 and 0.0894 for the train and test data sets, respectively for the removal of *g*. Hence, the input variables *f* and *g* are removed from the train and test data sets to produce a statistically significantly improved RMS error of 0.0451 and 0.0884 for the train and test data sets, respectively.

TABLE 3.2.11. GRNN Model Predictive Importance Results (*t*-critical = 2.345, α = 0.01, *df* = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
no variables omitted	0.0483	0.0925	-	-	19.08
a	0.0555	0.0942	4.244	2.2	18.13
b	0.0612	0.1052	7.823	16.5	18.23
c	0.0640	0.1104	9.076	23.2	18.30
d	0.0684	0.1134	13.651	27.1	18.17
e	0.0658	0.1164	21.395	31.0	18.29
f	0.0481	0.0891	3.759	0.0	18.34
g	0.0483	0.0894	3.886	0.0	18.29
non-cont. variables omitted	0.0451	0.0884	4.648	-	17.46

The casual importance measure of the input variables, documented in Table 3.2.12, has shown that varying *a* produces an increase in RMS error, 0.0674 and 0.0924 for the train and test data sets, respectively, while varying *e* produces the most significant increase in RMS error, 0.0811 and 0.1146 for the train and test data sets, respectively. It is also shown that the variables *b*, *c* and *d* each produced an increase in RMS error when varied. The alternative hypothesis is accepted for each *t*-test completed, confirming that the observed increase in error is statistically significant. The contribution of each input variable on the network prediction, using the measure of casual importance, is shown in Table 3.2.12.

TABLE 3.2.12. GRNN Model Casual Importance Results (*t*-critical = 2.345, α = 0.01, *df* = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		*t*-statistic	PerCon (%)	Comp. Time (s)
	Train	Test			
a	0.0674	0.0924	5.291	4.4	17.29
b	0.0753	0.1036	8.840	16.6	17.19
c	0.0793	0.1098	10.138	23.4	17.25
d	0.0814	0.1130	11.073	26.9	17.17
e	0.0811	0.1146	18.625	28.7	17.21

The percentage contribution of the input variables highlights *e* as the most significant input variable in the GRNN model influencing the prediction of *Y*. Further, it is shown that *d* is the next most significant input parameter, followed by *c*, then *b*, while *a* is least significant of the contributing input variables. In addition, it is shown that the variables *f* and *g* have no influence on the network predictions. It is shown that the measures of predictive and casual importance are in close agreement with the previous models regarding the ranking of importance of the input variables.

A study of computation time behaviour with changing network architecture has shown computation time to increase with increasing number of hidden layer nodes. Computation time is shown to be 0.21 seconds for 10 hidden layer nodes and σ equal to 0.2, increasing to 46.82 seconds for 1,000 hidden layer nodes. It is also shown that computation time decreases with decreasing number of input layer nodes and that the value assigned to σ has no appreciable influence on computation time.

3.3 QUANTITATIVE AND QUALITATIVE COMPARISON OF NEURAL NETWORK MODELS

While the characteristics and particularities of the developed neural network models have been discussed on an individual basis, it is now interesting to compare the results for the applied models. Moreover, a comparison of the neural network models with the completed MVRA is also interesting. Specifically, it is interesting to compare the error, ranking of input parameter importance and computation time associated with each model. While these measures provide a quantitative analysis of the developed neural network models, a qualitative analysis is also useful, completed by plotting the predicted shape of the mathematical function using the neural network models studied.

In order to determine the statistical significance of the difference in mean error for the applied models it is necessary to complete an ANOVA study. The null hypothesis is that the population means are equal, while the alternative hypothesis is that two or more of the population means are statistically significantly different. Treating the test data set error associated with each model as a population it follows that there are 7 populations of size 200 to be considered in the ANOVA study. The results of the ANOVA study are documented in Table 3.3.1, using a population number to

represent the error associated with each of the applied models, p_1 to p_7. A critical value for the F distribution, F_α, is obtained from an appropriate '*critical values of the F-distribution*' table [166]. In this instance F_α is equal to 2.1051, for a significance level of 0.05 and ν_1 and ν_2 equal to 6 and 1,393 respectively. It is shown that the ANOVA test statistic, documented as 9.2514, is higher in magnitude than the critical value of the test statistic, ie. $F > F_\alpha$. Therefore, the null hypothesis is rejected, hence, it is concluded that two or more population means are statistically significantly different.

TABLE 3.3.1. ANOVA Results for Comparison of Mean Error for Applied Neural Network Models and Multi-Variable Regression Analysis

ANOVA Statistics			
MST: 0.0361	*MSE*: 0.0039	*F*: 9.2514	F_α: 2.1051
Population	Mean Error		Variance
p_1 - (WH)	0.0221		0.0049
p_2 - (BP1)	-0.0094		0.0018
p_3 - (BP2)	0.0004		0.0022
p_4 - (RBF)	0.0192		0.0044
p_5 - (RBFKOH)	0.0114		0.0040
p_6 - (GRNN)	-0.0124		0.0051
p_7 - (MVRA)	0.0035		0.0054

While the ANOVA investigation has shown that two or more population means are unequal it provides no information on which particular population means are different. In order to determine this it is necessary to apply a statistical t-test to each paired population. It has been shown that the number of t-tests to complete for pairwise comparison of p populations is equal to $(p)(p\text{-}1)/2$, where p is equal to 7 in this instance, as there are 7 models applied, hence, there are 21 t-tests to complete. The results of these tests are documented in Table 3.3.2 using an index after the prefix t to indicate the paired populations for which the t-statistic value corresponds. For example, t_{12} corresponds to a t-statistic value for paired populations p_1 and p_2.

In order to evaluate the significance of the t-statistic values it is necessary to identify an appropriate t-critical value. Using a significance level of 0.01 and a degree of freedom value of 199, a t-critical value of 2.345 is appropriate. The null hypothesis for the t-test is that the means of the paired populations are equal, while the

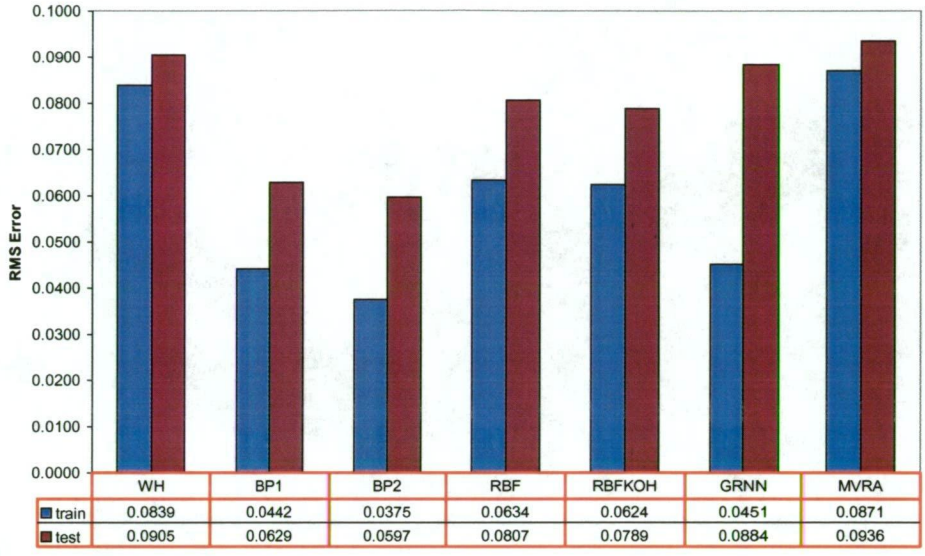
alternative hypothesis is that the means of the paired populations are unequal. If the *t*-statistic value is less than the *t*-critical value then the null hypothesis is accepted. However, it can be seen that the *t*-critical value is lower than the *t*-statistic value in each instance. Hence, each population mean is statistically significantly different, highlighting that the error associated with each model is statistically significantly different to the error associated with any other model.

TABLE 3.3.2. Statistical *t*-Test Results for Comparison of Mean Error for Applied Neural Network Models and Multi-Variable Regression Analysis

t_1 to t_7	t_2 to t_7	t_3 to t_7	t_4 to t_7	t_5 to t_7	t_6 to t_7
$t_{12} = 7.450$	$t_{23} = 3.835$	$t_{34} = 3.962$	$t_{45} = 3.254$	$t_{56} = 4.889$	$t_{67} = 4.438$
$t_{13} = 4.949$	$t_{24} = 7.407$	$t_{35} = 2.466$	$t_{46} = 7.014$	$t_{57} = 2.658$	
$t_{14} = 2.674$	$t_{25} = 5.946$	$t_{36} = 2.775$	$t_{47} = 3.165$		
$t_{15} = 3.258$	$t_{26} = 2.666$	$t_{37} = 2.669$			
$t_{16} = 9.397$	$t_{27} = 2.849$				
$t_{17} = 21.531$					

Figure 3.3.1 shows the train and test RMS error associated with the MVRA and each of the applied neural networks. The RMS error values shown represent the minimum error that each model was able to achieve with all redundant input variables removed. It can be seen that the minimum RMS error among the applied neural networks was achieved using the BP2 network, shown to be 0.0375 and 0.0597 for the train and test data sets, respectively. On the other hand, the highest prediction error among the applied neural networks was associated with the WH network, shown to be 0.0839 and 0.0905 for the train and test data sets, respectively. The BP1 network was found to produce comparable error to the BP2 network, shown to be 0.0442 and 0.0629 for the train and test data sets, respectively. In addition, the RBF network, using function centres selected randomly from the training data produced slightly higher error, shown to be 0.0634 and 0.0807 for the train and test data sets, respectively. On the other hand, the RBFKOH network, with function centres determined using a Kohonen neural network for clustering similar training data patterns, produced a slightly lower RMS error than the RBF with randomly selected function centres, shown to be 0.0624 and 0.0789 for the train and test data sets, respectively. The GRNN model has shown the second highest prediction error of the neural network models studied, shown to be 0.0451 and 0.0884 for the train and test data sets, respectively. In addition, it is shown that a train and test RMS error of 0.0871 and

0.0936, respectively, is obtained using the MVRA, which is higher than that achieved using any of the applied neural networks.



	WH	BP1	BP2	RBF	RBFKOH	GRNN	MVRA
train	0.0839	0.0442	0.0375	0.0634	0.0624	0.0451	0.0871
test	0.0905	0.0629	0.0597	0.0807	0.0789	0.0884	0.0936

Fig. 3.3.1. Comparison of RMS Error Associated with Developed Neural Network Models and Multi-Variable Regression Analysis

While it is interesting to compare RMS error associated with each of the applied neural networks it is also necessary to study the features of the training data used by each of the applied models. Hence, Table 3.3.3 highlights the percentage contribution of each of the input variables for each of the applied neural networks, using predictive importance as a measure in the first instance. It is shown that there is some difference in input variable importance, using percentage contribution as an indication, for the different neural network models. It is shown that the importance of variable *a* ranges from a maximum of 12.9% using the BP2 network, to a minimum of 2.0% using the WH network. While *b* is shown to have different levels of importance, a maximum of 17.9% using the BP1 network and a minimum of 13.2 using the WH network, the range of values is shown to be lower for *b*, compared to *a*. In addition, the range of *c* and *d* are shown to be a maximum and minimum of 23.2% and 19.7%, respectively, for the GRNN and RBFKOH networks, respectively, for *c*, and a maximum and minimum of 28.5% and 24.4%, respectively, for the WH and BP1 networks, respectively, for *d*. Further, *e* is shown to have a maximum percentage contribution of 33.9% using the WH model and a minimum percentage contribution of 25.7% using the BP2 network. Nevertheless, each of the studied

neural network models has shown *f* and *g* to have zero influence on the output variable.

TABLE 3.3.3. Percentage Contribution of Input Variables using Predictive Importance Technique

Input Variable	Percentage Contribution of Input Variables (%)					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
a	2.0	7.8	12.9	10.5	5.3	2.2
b	13.2	17.9	14.5	16.7	15.2	16.5
c	22.3	20.9	22.5	20.8	19.7	23.2
d	28.5	26.6	24.4	25.1	26.1	27.1
e	33.9	26.8	25.7	26.8	33.7	31.0
f	0.0	0.0	0.0	0.0	0.0	0.0
g	0.0	0.0	0.0	0.0	0.0	0.0

A comparison of ranking of importance of the input variables using casual importance as a measure, as shown in Table 3.3.4, has shown similar results to the predictive importance measure. It can be seen that while the percentage contribution of the input variables is different for each of the neural network models studied, the casual importance results for each model compare well with the predictive importance results for each model. While there are noted differences among the developed neural network models regarding the percentage contribution of the input variables, it is shown that the order of ranking exhibited by each model is the same in each instance. Input parameter *e* is highlighted in each instance as the most significant parameter in the mathematical function, while *a* is shown to be the least significant. Further, parameter *d* is highlighted as the second most significant parameter, followed by *c* and then *b*. For each of the applied models, the parameters *f* and *g* were shown to have no influence on the prediction of *Y*.

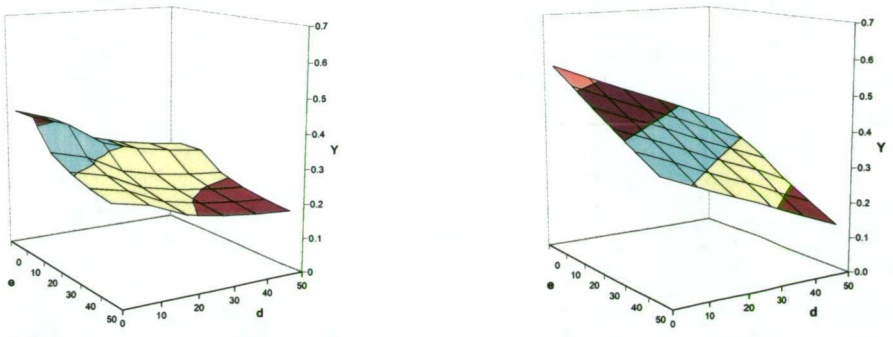
TABLE 3.3.4. Percentage Contribution of Input Variables using Casual Importance Technique

Input Variable	Percentage Contribution of Input Variables (%)					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
a	10.2	7.2	11.7	12.7	8.6	4.4
b	14.3	17.8	15.6	17.1	12.6	16.6
c	21.3	22.3	22.2	21.9	21.4	23.4
d	26.4	26.3	24.2	23.5	26.2	26.9
e	27.7	26.4	26.2	24.8	31.1	28.7
f	-	-	-	-	-	-
g	-	-	-	-	-	-

While it is interesting to compare the RMS error for the MVRA and the applied neural network models, it is also interesting to compare the computation time associated with the applied models. It is important to note that the computation time considered is the time required by each model to develop a model of the mathematical function with all redundant input variables removed. While the MVRA has been shown to produce the highest RMS error, it is shown that the computation time associated with the MVRA is approximately 3.90 seconds, which is significantly lower than that achieved using the neural network models. It is shown that the WH network requires a relatively low computation time, 109.26 seconds, while the BP1 and BP2 networks require computation times of 822.72 and 1,182.46 seconds, respectively. Further, the RBF network is shown to require a computation time of 992.84 seconds, which is significantly lower than that necessary for the RBFKOH model, shown to be 2,645.60 seconds. It can be seen that the lowest computation time for the applied neural networks is achieved using the GRNN model, shown to be 17.46 seconds. It is shown from a study of computation time associated with each of the applied neural networks that computation time is significantly influenced by the complexity of the network architecture, the activation function used in the processing nodes of the network, the procedure used to specify the connection weights in the network and further, the process used to update the network weights during training. While previous discussion has detailed the trend of increasing computation time with increasing number of network nodes and changing activation functions, a comparison of the GRNN model with the remaining networks highlight the behaviour of computation time with the procedure used to set and update the network weights. It is shown that the GRNN model, which has all network weights set using the training data and does not have weights updated during training, requires a very low computation time. However, the WH, BP1, BP2, RBF and RBFKOH networks have higher computation times, in comparison, due to the delta rule weight update procedure used in each of the models. Moreover, a comparison of the GRNN and RBF network, for 10 hidden layer nodes, 7 inputs and σ equal to 0.1, has shown a computation time of 225.64 seconds is required for the RBF model. While the GRNN model requires a computation time of only 0.22 seconds, which is approximately 0.1% of the computation time required by the RBF network. This significantly lower computation time can be attributed to the fact that the network weights are not updated during training of the GRNN model, but rather set at

permanent values from a single pass of the training data. This is the nature of the algorithm used in this particular network model.

While it is useful to complete a quantitative comparison of the developed neural network models, it is also interesting to complete a qualitative comparison. This is achieved by comparing the predicted shape of the mathematical function for each network against the shape of the mathematical function plotted using the given formula. The shape of the mathematical function is shown in Figure 3.3.2(a), calculated using the relationship shown in Equation 3.1.4. To allow graphical representation of the mathematical function, it is necessary to set some of the source variables equal to a constant value while others are varied. It has been shown as a result of the MVRA and neural network modelling that the source variables e and d are of higher influence on the response variable than other source variables. Hence, it follows that a plot of the response variable, Y, against the source variables e and d will provide the greatest definition in the shape of the function. While e and d are increased over the range of 0 to 50, the remaining source variables are maintained at a constant mean value of 25, with the corresponding value of Y calculated accordingly. While the function is modelled over the range 0 to 50 for increasing values of e and d, it is important to note that while the mathematical function has been shown to be discontinuous for $d = 4$ and $e = 5$, these discontinuities are not featured on the model in order to highlight more sharply the shape of the function over its continuous range. However, it is important to note that the function has asymptotes at $d = 4$ and $e = 5$. For comparison, the predicted shape of the mathematical function using the MVRA is shown in Figure 3.3.2(b). It can be seen that the predicted shape of the mathematical function using the MVRA is significantly different to the actual shape of the specified mathematical function. While the shape of the specified mathematical function is of non-linear form, the predicted shape of the mathematical function using the MVRA is shown to be of linear form, attributed to the linear form of the regression equation.



(a) (b)

Fig. 3.3.2. Shape of Mathematical Function Calculated using (a) Specified Mathematical Equation, and (b) MVRA

The predicted shapes of the mathematical function for each of the neural network models studied are shown in Figures 3.3.3(a) to (f). For the neural network models, it can be seen that the BP2 network most closely represents the actual shape of the mathematical function, as shown in Figure 3.3.3(c). However, the BP1 network, as shown in Figure 3.3.3(b), and the GRNN model, as shown in Figure 3.3.3(f), also closely represent the actual shape of the specified mathematical function. In addition, while the RBF and RBFKOH networks yield similar predicted shapes, Figures 3.3.3(d) and (e), respectively, they are not as accurate as the backpropagation and GRNN models. In addition, the predicted shape of the mathematical function using the WH model, Figure 3.3.3(a), while relatively similar to the actual shape of the function, is least accurate of the studied network models.

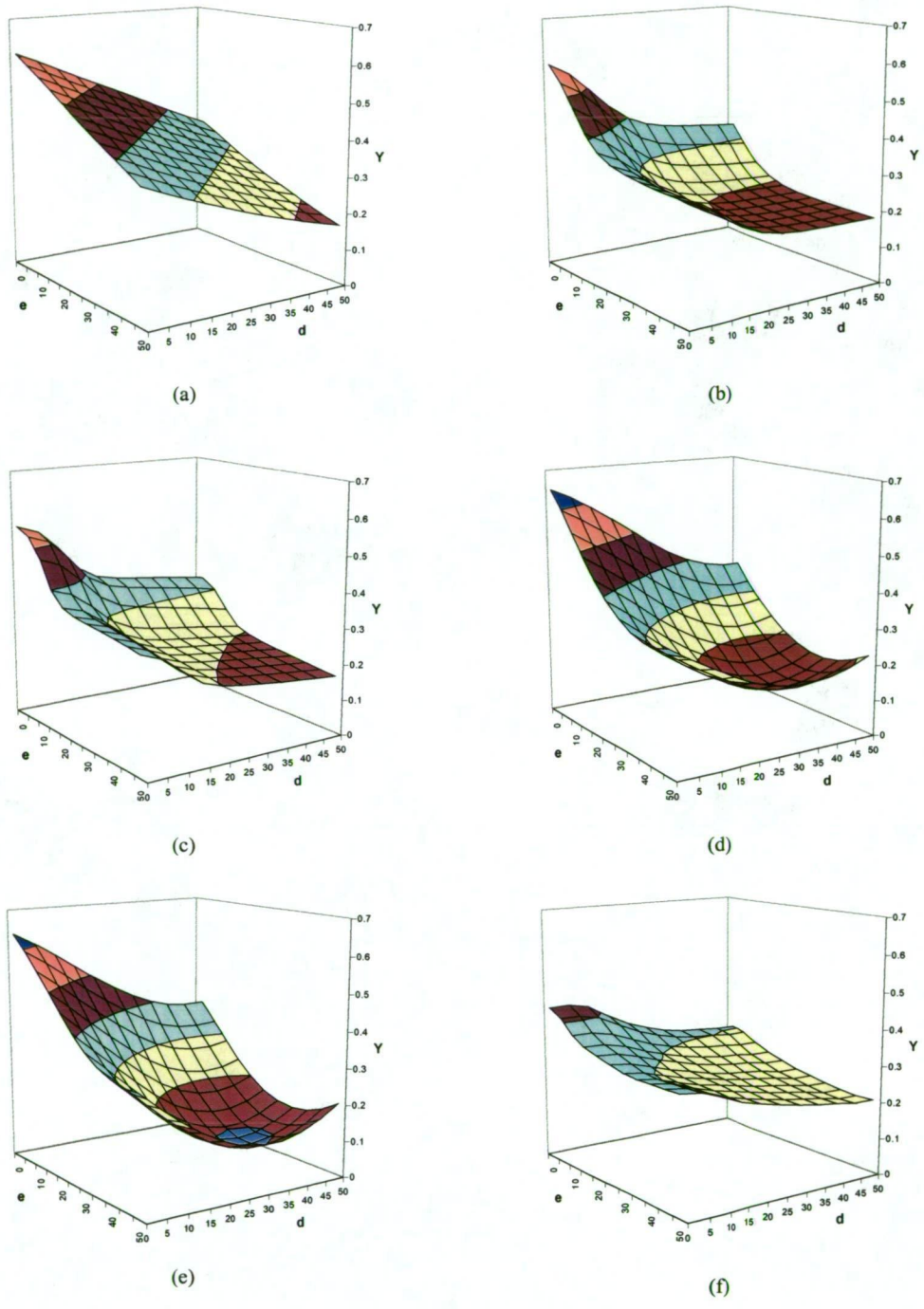


Fig. 3.3.3. Predicted Shape of Mathematical Function using Applied Neural Network Models (a) WH, (b) BP1, (c) BP2, (d) RBF, (e) RBFKOH, and (f) GRNN.

3.4 CONCLUDING REMARKS

Based on the quantitative and qualitative measures used as a part of this sensitivity analysis to study the performance of the applied neural network models, it has been shown that each of the studied models were capable of developing a relatively accurate model of the specified mathematical function. This result confirms the suitability of the selected neural networks to be applied in the aluminium smelting

industry for process modelling and estimation. It is shown that the BP2 network produced the most accurate model of the mathematical function, the computation time associated with this particular model is shown to be significantly higher compared to some of the other models studied. However, a static model of the mathematical function was required in this instance, making computation time a low priority for network selection. However, for neural network models that are required to train in real-time and make real-time predictions, computation time can become a significant feature for network selection. In addition, the percentage contribution of the input variables can also be a significant consideration when selecting a neural network for a particular application. Hence, when selecting a neural network for a particular application to achieve economic benefit it is important to have a selection technique that considers the following:

i). network model accuracy and cost associated with accuracy of network prediction

ii). input variables required for network model and cost associated with obtaining data for required variables, and

iii). computation time required by the network model to converge to a solution and the cost associated with this time

While a suitable selection technique is not discussed here it is useful to note that an optimisation methodology for selecting a neural network based on the specified selection criteria will be detailed in a later chapter.

The sensitivity analysis completed here has been useful to investigate the behaviour of the developed neural network models. Using the specified mathematical function, with a known underlying relationship, it has been possible to study the accuracy of each of the neural network models and the associated computation time required for accurate modelling. Moreover, this sensitivity analysis has shown the predictive and casual importance techniques to be useful methodologies for determining the relative importance of input variables. The results obtained from this investigation give confidence in applying these neural network models and techniques for determining input variable importance to practical applications where the underlying relationships among process parameters are unknown. While the results of applying these neural

network models to particular applications in the aluminium smelting industry are discussed in a later chapter, it is necessary to document the particularities of the studied applications. Hence, the following chapter discusses the particularities of each application considered and provides a rationale for using neural networks for modelling and estimation in the aluminium smelting industry.

CHAPTER FOUR

Incorporating Intelligent Control into the Aluminium Smelting Process

4.1 INTRODUCTION TO INTELLIGENT CONTROL STRATEGY

Opportunities exist within the aluminium smelting industry to incorporate advanced process control methodologies to achieve productivity improvements. Phenomenological modelling in the aluminium industry typically requires complex models as it involves large scale, multi-stage, multi-variable, non-linear and dynamic operation. Moreover, relationships among process parameters in the aluminium smelting industry are generally highly dimensional and non-linear, with the presence of noise in most instances. Mathematical modelling in the aluminium industry is particularly difficult due to process instability, high non-linearity and limited understanding of the first principles of the Hall-Heroult process. However, neural networks offer an alternative modelling strategy that has particular suitability to the aluminium smelting industry. Due to the development in recent years of intelligent sensors and analytical equipment to provide critical process information, substantial historical data is readily available on the smelter knowledge base. This information is fundamental for training neural networks to model particular aspects of the aluminium smelting process. A significant advantage of neural network modelling in the aluminium industry is that it is not necessary to understand the complex first principles of the Hall-Heroult process or make simplifying assumptions regarding process behaviour. Hence, neural networks are applied as intelligent decision making tools in the aluminium smelting industry as a part of this work. It is necessary then to discuss the particularities of the applications considered and highlight their suitability to neural network modelling, with the objective of achieving economic benefit as a result. Having completed a brief review of the Hall-Heroult process for the production of aluminium in a previous chapter, this particular chapter deals with more specific areas of aluminium production that are studied as neural network

modelling applications. For each of the practical applications considered here it is useful to provide some relevant theory in order to highlight the significance of incorporating neural network modelling in each instance. In addition, the associated process parameters that are considered as neural network inputs in each instance are detailed. It is useful to provide some theory of these selected inputs in this chapter, as they are particularly relevant to the practical applications studied and in some instances require considerable discussion. Moreover, for each application it is necessary to determine those process parameters that have some relationship with the process under investigation and are available as network inputs. However, it is important to note that minimising the number of network inputs at this stage of the modelling process is not critical. Rather, the parameters selected as network inputs at this stage of the modelling process should include all parameters that are considered to have some relationship with the network output parameters, regardless of the extent of the relationship. It has been shown in the previous chapter that there are specific techniques available that can be used to identify contributing parameters in a neural network model and consequently, eliminate non-contributing parameters. Therefore, to achieve maximum modelling performance it is preferable to eliminate input parameters during the neural network modelling phase, rather than during the initial input variable selection stage. While a particular process parameter may be considered to have only a minor influence on the accuracy of the neural network, modelling may show that particular parameter to indeed have a significant influence on the network prediction accuracy. Hence, to eliminate that particular parameter before network modelling commences would disadvantage the accuracy of the network prediction. In addition, it is important to note that the selection of network inputs in each instance is completed by observing graphically the behaviour of various process parameters with changing process conditions. In addition, communications with experienced process personnel identified further process parameters that should be considered as neural network inputs in each instance.

4.2 APPLICATION OF NEURAL NETWORKS TO PREDICT ELECTROLYTE ADDITIVES

The application being scoped in this instance involves an improved methodology for scheduling electrolyte additives to industrial reduction cells at CABBL. Due to the dynamic behaviour of the Hall-Heroult process, the quantity of electrolyte additive to

add to the reduction cell is a complex decision involving a number of specific process parameters. It is shown in the following section that the technique applied at CABBL to determine a particular quantity of electrolyte additive for each reduction cell is based on these process parameters. However, there is an opportunity to improve the existing electrolyte additive scheduling methodology by applying a neural network to model this part of the Hall-Heroult process. It is shown that the quantity of electrolyte additive scheduled for each reduction cell plays an important role in determining the efficiency of the aluminium smelting process. In particular, the application of neural networks for the prediction of electrolyte additives has been investigated by the author with the findings published in several international refereed conference proceedings [171-175], where the suitability of neural networks for the application have been well accepted by researchers in the field. In order for the reader to understand the application in this instance, it is useful to detail the importance and characteristics of electrolyte additives.

4.2.1 Importance and Characteristics of Electrolyte Additives

It should be noted that the electrolyte additives scheduled regularly for industrial reduction cells include aluminium fluoride, AlF_3, and soda ash, Na_2CO_3. Further, AlF_3 has been introduced in a previous chapter as the most common electrolyte additive scheduled to reduction cells at CABBL. The need for regular additions of electrolyte additives to the bath composition arises from the various loss mechanisms of the additives and variations in bath superheat, due in part to changing liquidus temperature [27]. Nevertheless, evolution of AlF_3 from the reduction cell occurs as a byproduct of the aluminium smelting process through various mechanisms. Fluorides within the electrolyte are extremely volatile and therefore easily lost from the cell. Hence, regular additions of AlF_3 are required in order to maintain the bath chemistry. It has been noted in a previous chapter that AlF_3 is added to the reduction cell for the beneficial affect of lowering the liquidus temperature of the electrolyte, leading to significant improvements in current efficiency. This is shown by considering the relationships between AlF_3 and bath temperature and bath temperature and current efficiency in reduction cells at CABBL that have been studied and documented by Stevens *et al* [176]. The significant findings from this work are shown in Figures 4.2.1 to 4.2.3. Firstly, Figure 4.2.1 shows that there exists a definite relationship between AlF_3 content and bath temperature. This is confirmed by the correlation

coefficient, *r*, which is shown to be -0.896 in this instance. Further, the regression line drawn on the graph shows that bath temperature decreases with increasing percent excess AlF_3 content.

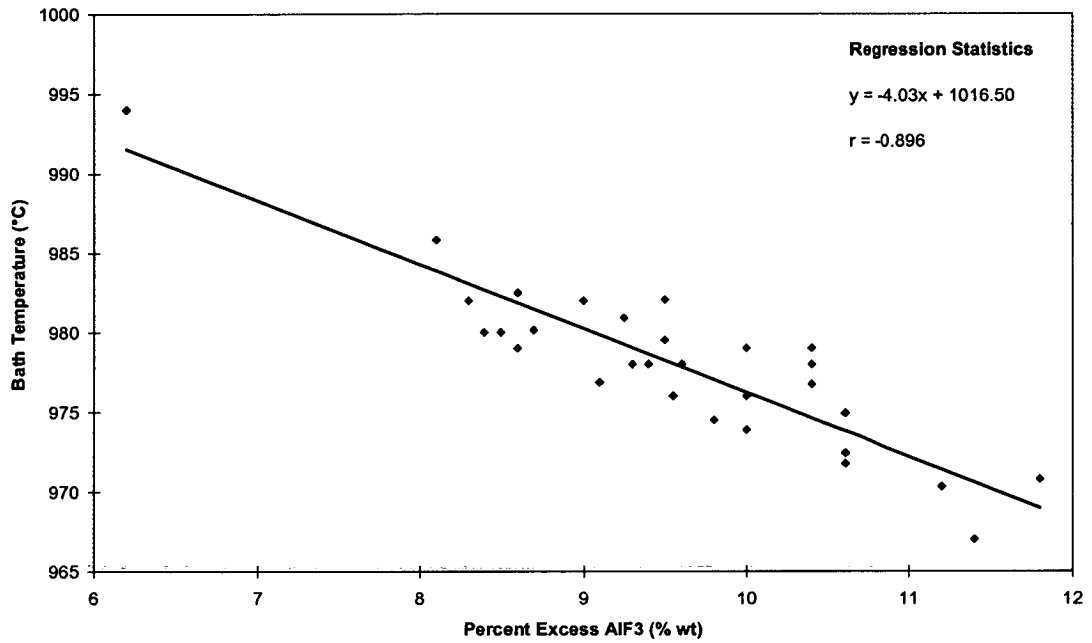


Fig. 4.2.1. Correlation Between Average Bath Temperature and Average Percent Excess AlF_3 [176]

Figure 4.2.2 shows that there is also a high correlation between current efficiency and bath temperature, again confirmed by a correlation coefficient of -0.893. It is shown that current efficiency is high for bath temperatures of approximately 968.0°C and further, that current efficiency decreases with increasing bath temperature. In addition, a graphical analysis has shown that a 1.0°C increase in bath temperature results in a decrease in current efficiency of 0.45%, over the range of 968.0 to 993.0°C. This is given by the coefficient of the regression line equation. While it is shown that increasing AlF_3 content decreases bath temperature and further, that decreasing bath temperature increases current efficiency, Stevens *et al* have also documented a direct relationship between AlF_3 and current efficiency, as shown in Figure 4.2.3. It can be seen that there is a strong correlation between percent excess AlF_3 and current efficiency in the reduction cell, confirmed by the correlation coefficient of 0.942. It is useful to note that a graphical analysis has shown that an

increase of 1.0 percent by weight excess AlF_3 yields an increase in current efficiency of 1.66%, over the range of 6.2 to 11.8 percent by weight excess AlF_3.

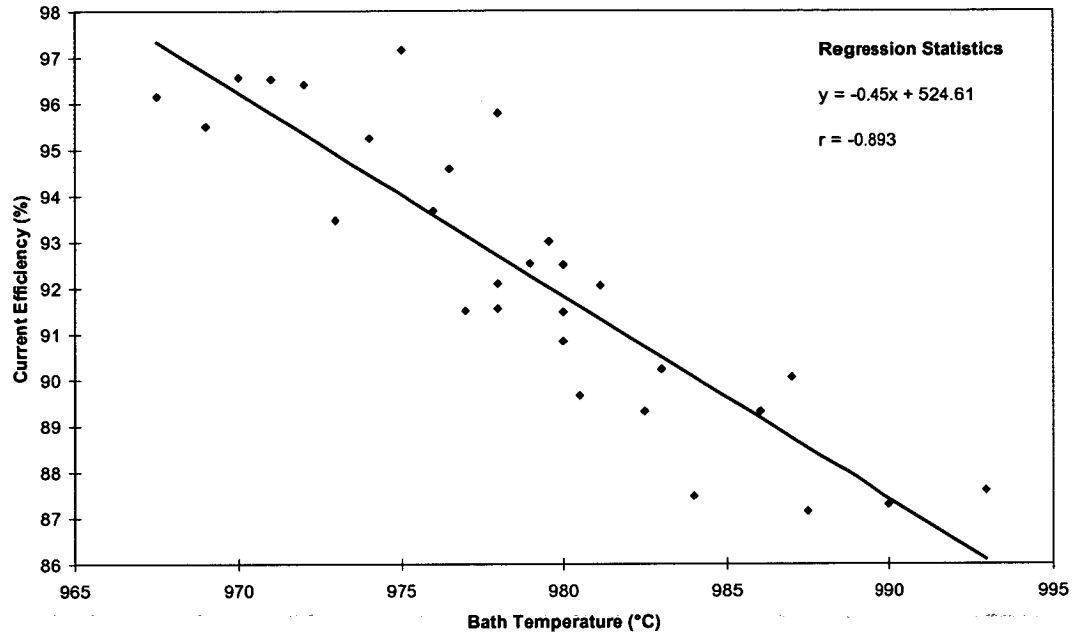


Fig. 4.2.2. Correlation Between Average Bath Temperature and Current Efficiency [176]

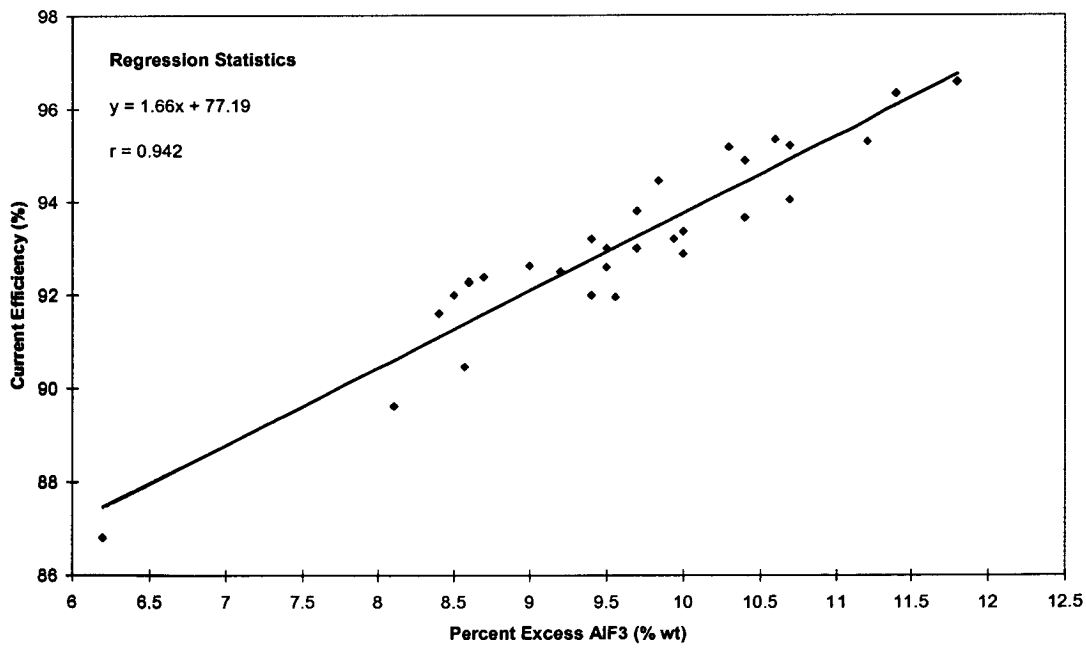


Fig. 4.2.3. Correlation Between Average Percent Excess AlF_3 and Current Efficiency [176]

In addition to lowering bath temperature and increasing current efficiency, it is important to note that AlF_3 is also beneficial in reducing the solubility of metals in the electrolyte and for lowering the bath density, both of which contribute towards improving process efficiency. In contrast, Na_2CO_3 is also a necessary electrolyte additive for efficient reduction cell operation as it has the property of increasing the liquidus temperature of the electrolyte, which is sometimes necessary if the electrolyte temperature has decreased below some predetermined minimum control limit. However, for economic reasons it is beneficial to maintain bath temperature above the predetermined minimum and avoid the requirement for additions of Na_2CO_3. Na_2CO_3 is a costly element and has a stoichiometric ratio with AlF_3 of approximately unity, meaning that for every kilogram of Na_2CO_3 used approximately 1.0 kilogram of AlF_3 is wasted. Hence, it is economically beneficial to add a lower quantity of AlF_3 to the reduction cell in the first instance, rather than add a large quantity of AlF_3 compensated by a subsequent addition of Na_2CO_3.

In considering the importance of electrolyte additives within the reduction cell, it is important to note that at CABBL a particular control range for bath temperature is specified, as shown in Figure 4.2.4. It can be seen that the lower and upper control limits for bath temperature are 950.0 and 980.0°C, respectively [17].

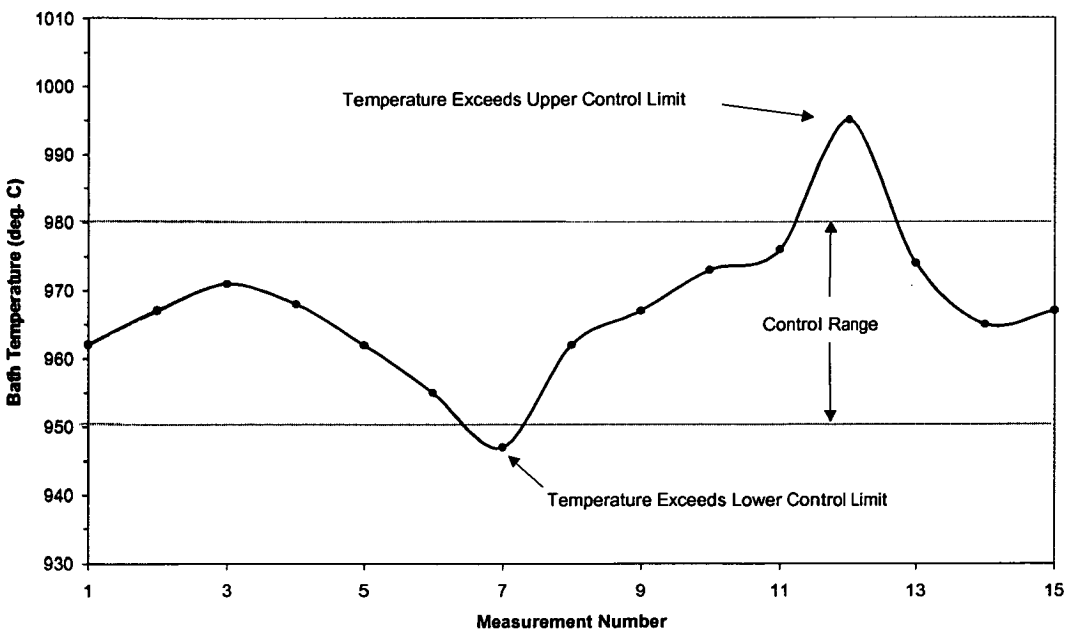


Fig. 4.2.4. Control Chart Highlighting Lower and Upper Bath Temperature Control Limits at CABBL

The primary criteria for bath temperature control at CABBL is to keep the bath molten in order to maintain stable cell operation, leading to reduced thermal cycling and high current efficiency, facilitate fast aluminium dissolution and avoid operating in a two-phase (liquid-solid slurry) system [177]. Low bath temperature is generally attributed to insufficient power input, poor alumina cover, high metal depth and high quantities of undissolved alumina in the bath [178]. If the bath temperature is allowed to decrease below the lower control limit of 950.0°C some of the following problems can occur as a consequence:

i). the solubility of alumina feed is reduced, which can result in sludge and freezing of bath on the cathode, leading to a reduction in process efficiency
ii). the frequency of anode effects can increase, which cause major instabilities in the reduction cell and significantly reduce current efficiency, and
iii). routine operations, such as anode changing, are made more difficult due to the increased presence of solid lumps within the electrolyte and harder electrolyte crust

While the disadvantages of low bath temperature are highlighted, it is also important to note that high bath temperature also has a detrimental effect on the aluminium smelting process. High bath temperature is generally attributed to low bath and metal levels, insufficient AlF_3 additions and excessive voltage being applied to the cell. If bath temperature increases above the upper control limit of 980.0°C some of the following problems can occur as a consequence:

i). significant reduction in process efficiency
ii). ledge melt may occur, which reduces the protection of the cell side walls from cryolite exposure, which can result in damage to the reduction cell lining and ultimately cause early cell failure
iii). crust thickness can be reduced, thereby producing heat losses and volatile gasses, such as toxic fluorides, and
iv). increase in risk of new anodes shattering while being placed in the reduction cell

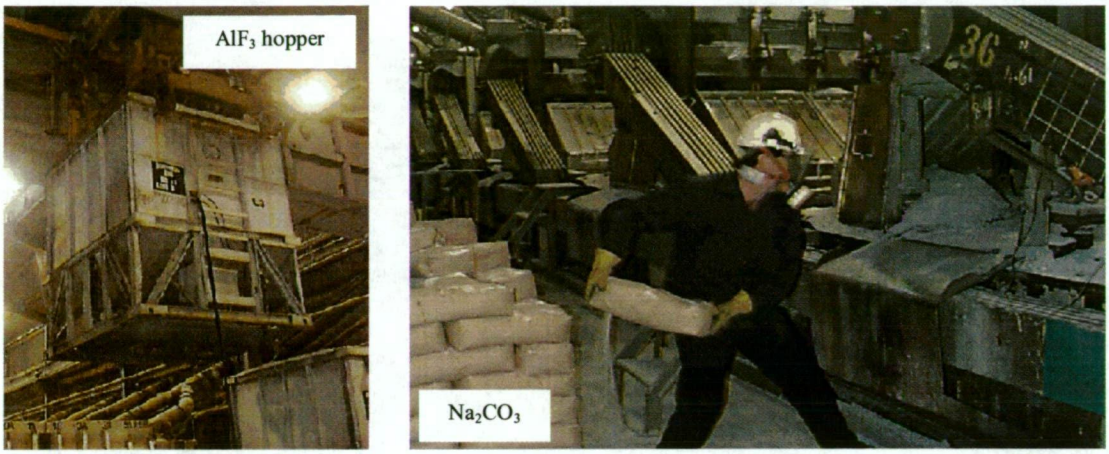
At CABBL a bath temperature of 965.0°C is targeted as optimum for cell operation, with minimum deviation from this temperature desirable for stable cell operation. This particular bath temperature has been determined through significant theoretical and experimental investigations during process establishment. While lower operating temperatures have been reported at smelters elsewhere around the world [179, 180] and are beneficial for achieving higher current efficiency, a bath temperature of 965.0°C has been identified as the lowest target temperature at which CABBL can operate without the onset of undesirable process occurrences, such as those listed previously, given that bath temperature standard deviation is typically in the range of 8.0 to 10.0°C. While higher bath temperatures increase the solubility of alumina, leading to better dissolution of alumina, current efficiency decreases and anode deterioration is significantly increased. On the other hand, lower bath temperatures can yield higher current efficiency, however, poor dissolution of alumina and crystallisation of alumina on the carbon cathode, leading to poor heat balance within the reduction cell, occurs at lower bath temperatures. Hence, it is important to note that a bath temperature of 965.0°C is required at CABBL while minimum deviation from this temperature is necessary for achieving high current efficiency and improved operating conditions within the reduction cell. It has been shown that variability over time is the major driver of loss of current efficiency in otherwise well designed reduction cells [23]. A number of studies of losses in current efficiency have been undertaken by Stevens *et al* [176], highlighting that the average current efficiency will decrease as the degree of variation increases, even if the same average conditions are maintained in the reduction cell. Industry current efficiencies for most technologies are in the range of 90.0 to 95.0% and none is higher than 95.5% over a sustained period [176]. However, Stevens *et al* have shown conclusively in their work that a current efficiency of 97.0 to 98.0% is possible if average temperature, electrolyte composition and other conditions are maintained in the cell.

4.2.2 Existing Technique for Scheduling AlF_3 and Na_2CO_3 Additions to the Reduction Cell at CABBL

The procedure described here details the existing process for adding AlF_3 and Na_2CO_3 to reduction cells at CABBL. This is necessary to highlight the existing technique used for the addition of these important elements to the electrolyte, leading to a justification for the replacement of the existing technique using neural network

modelling. In particular, addition of AlF_3 and Na_2CO_3 to the industrial reduction cell at CABBL is essentially a three-stage process, consisting of identification of existing percent excess AlF_3 in the electrolyte, calculation of addition of AlF_3 or Na_2CO_3 based on percent excess AlF_3 deviation from the mean and addition of the calculated amount of AlF_3 or Na_2CO_3 to the reduction cell. In order to determine the existing percent excess AlF_3 in the electrolyte of a reduction cell a multiple regression analysis is used. It has been noted that multiple regression is an attempt to model or explain a process that generates data by deriving a certain type of linear relationship from that data and that it is a statistical technique that assesses the relationship between one dependent variable and any number of independent variables. Regression techniques can be applied to a data set in which the independent variables are correlated with one another and with the dependent variable to varying degrees [181, 182]. In this instance, the regression analysis uses bath temperature and bath resistivity as independent parameters for predicting the dependent parameter, percent excess AlF_3. As noted, bath temperature is significantly influenced by electrolyte chemistry, hence, it is a useful indicator of percent excess AlF_3 present in the electrolyte. Bath temperature is measured per cell every 4 days, using high temperature thermocouples and appropriate data acquisition. On the other hand, bath resistivity is a measure of the electrical conductivity of the electrolyte and is also significantly influenced by the electrolyte chemistry. Bath resistivity is measured per cell every 10.0 minutes using an automated process within the reduction cell, with readings being stored in both a short- and long-term database for referencing. With the automated system, a daily average of bath resistivity for each cell is calculated. The average bath resistivity is then used in the regression analysis for predicting percent excess AlF_3. The estimate of percent excess AlF_3 from the regression analysis is used to calculate the addition mass of AlF_3 or Na_2CO_3 so that the cell can be maintained or returned to a target electrolyte composition in order to achieve a mean bath temperature of 965.0°C. The amount of AlF_3 or Na_2CO_3 to add to a cell is a combination of the difference between the estimated percent excess AlF_3 and the target electrolyte concentration and the default addition mass. AlF_3 is commonly added to industrial reduction cells at CABBL using a procedure that involves the use of a storage hopper to make precise additions to the reduction cell, as shown in Figure 4.2.5(a), while Na_2CO_3 is added using a manual technique, as shown in Figure 4.2.5(b). While Na_2CO_3 is an important electrolyte additive its usage is only

approximately 5.0% of AlF_3 usage, hence, due to the low frequency of additions of Na_2CO_3 it is economically more viable to use the manual addition technique.



(a) (b)

Fig. 4.2.5. Method of Electrolyte Additive Addition to Reduction Cell (a) AlF_3, and (b) Na_2CO_3

4.2.3 Rationale for Electrolyte Additive Prediction Using Neural Network Modelling

In considering the rationale for neural network modelling in this instance it is important to reiterate that additions of Na_2CO_3 to the reduction cell are typically to compensate for the addition of too high a quantity of AlF_3. Hence, the addition of Na_2CO_3 may be avoided if the electrolyte additive technique accurately schedules the correct quantity of AlF_3 in the first instance. Hence, the quantity of Na_2CO_3 added for this purpose is wasted material. However, in some instances the addition of Na_2CO_3 is required due to reasons other than the addition of too high a quantity of AlF_3, such as newly installed cells that require Na_2CO_3 to compensate for the high absorption rate of Na_2CO_3 into the cathode during the first 30 days of cell life. In this instance, the addition of Na_2CO_3 is required to compensate for the absorbed Na_2CO_3, such that cryolite is formed. A study of electrolyte additive consumption at CABBL has shown that the approximately 80.0% of Na_2CO_3 usage is classed as waste, to compensate for high AlF_3 additions, while the remaining 20.0% is required for newly installed cells. Hence, there is an opportunity to reduce production costs through a reduction in Na_2CO_3 wastage. A typical consumption value for Na_2CO_3 is 215.0 tonnes per annum, based on 80.0% wastage, this equates to 172.0 tones of Na_2CO_3 wastage per

annum. Further, an improved scheduling technique is expected to result in a reduction of wasted AlF_3, which is the quantity of AlF_3 added to the cell for which Na_2CO_3 is added to compensate. Based on a stoichiometric ratio of unity, 172.0 tonnes of Na_2CO_3 wastage per annum results in 172.0 tonnes of AlF_3 wastage per annum. Hence, the economic benefit of implementing neural networks to accurately predict electrolyte additives is through reduced AlF_3 and Na_2CO_3 wastage. In addition, there is a potential for improvements in current efficiency through accurate scheduling of electrolyte additives and hence a reduction in bath temperature variation, which subsequently results in lower production costs. In addition, it is noted that reduced bath temperature variation yields reduced thermal cycling of the cathode, resulting in extended cell life, for which there is also an economic benefit.

4.2.4 Specification of Neural Network Inputs for Electrolyte Additive Prediction Application

While the parameters used as network inputs for this particular application are measured parameters of the Hall-Heroult process at CABBL, it is important to note that a critical consideration for the success of neural network modelling is to have suitable and sufficient network inputs. The inputs individually must play some role in improving the network performance and combined must be sufficient enough to represent the process behaviour to allow maximum accuracy in network predictions. Hence, it is important to note that the parameters specified here include any process variables that are considered to provide at least some relationship with bath chemistry. While the network inputs selected here may not contribute to the network prediction, it has been noted that is not until neural network modelling is commenced that this conclusion can be drawn. Hence, it is important here to include all parameters that may have an influence on the network prediction of electrolyte additive quantity and eliminate those parameters from the model that do not contribute after neural network modelling is evaluated. In particular, the process parameters selected as neural network inputs for this application are shown in Figure 4.2.6, using an arbitrary neural network architecture.

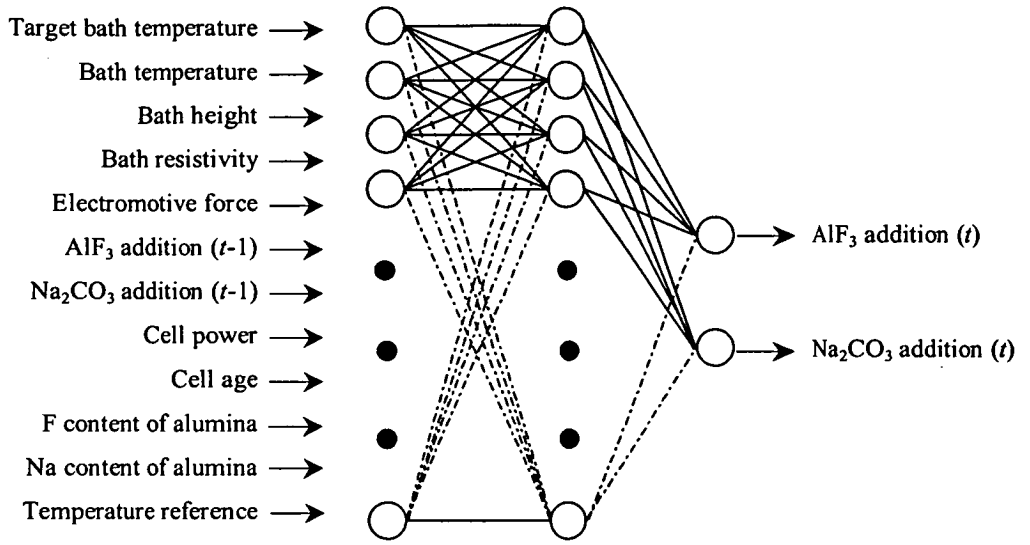


Fig. 4.2.6. Illustration of Neural Network Model for Electrolyte Additive Prediction Application

The following section gives a brief description of each process parameter used in this instance and provides a justification for its selection as a network input for this particular application. Further, the selected process variables are noted as appropriate in Figure 4.2.7 to highlight their role in the reduction cell.

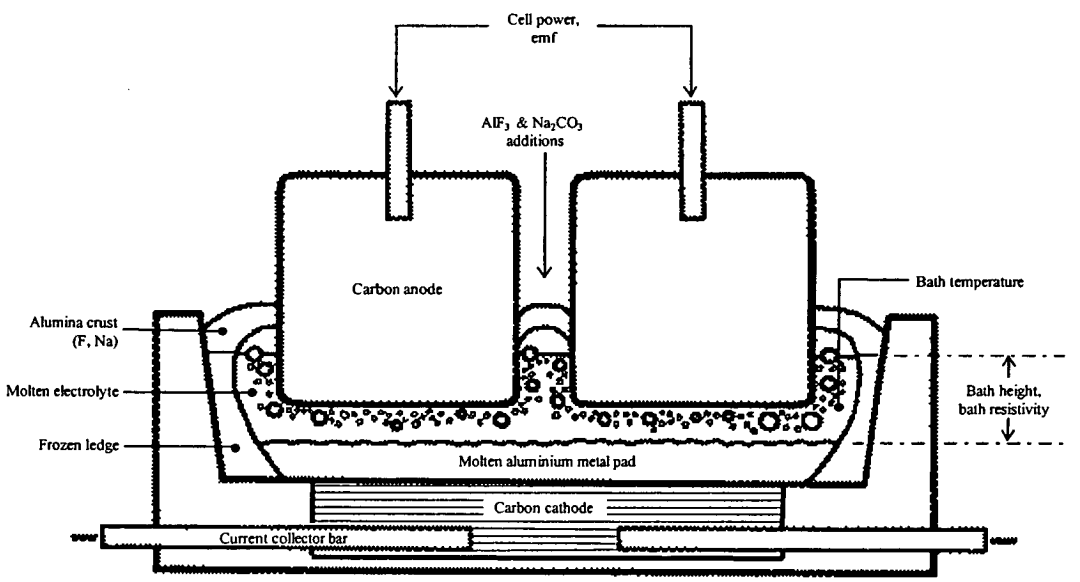


Fig. 4.2.7. Cross-Section of Reduction Cell Indicating Process Parameters used as Neural Network Inputs for Electrolyte Additive Prediction Application

1. Target bath temperature - Target bath temperature as a network input specifies the temperature that is desired for the bath after an electrolyte additive addition is completed. In this instance the target temperature for the electrolyte is 965.0°C, for reasons discussed previously. With the specification of the target bath temperature, the neural network will estimate an addition with the consideration that a bath temperature of 965.0°C is desired following an electrolyte additive addition.

2. Bath temperature - Bath temperature is an important network input as it specifies the temperature of the electrolyte in the reduction cell. It has been shown that bath temperature is significantly influenced by electrolyte chemistry, more specifically, bath temperature decreases and increases with increasing AlF_3 and Na_2CO_3 content, respectively. Hence, it is critical that bath temperature be included as a network input as it provides a substantial indication of the percent excess AlF_3 present in the electrolyte. Bath temperature is measured manually by inserting the tip of a K type thermocouple approximately 25.0 to 75.0mm below the electrolyte surface. The thermocouple is inserted at a shallow angle and left resting in the electrolyte, as shown in Figure 4.2.8. A digital gauge connected to the thermocouple is monitored to observe the corresponding bath temperature. Each completed bath temperature measurement is recorded and entered into the smelter knowledge base. Typical values of bath temperature are in the range 935.0 to 995.0°C.

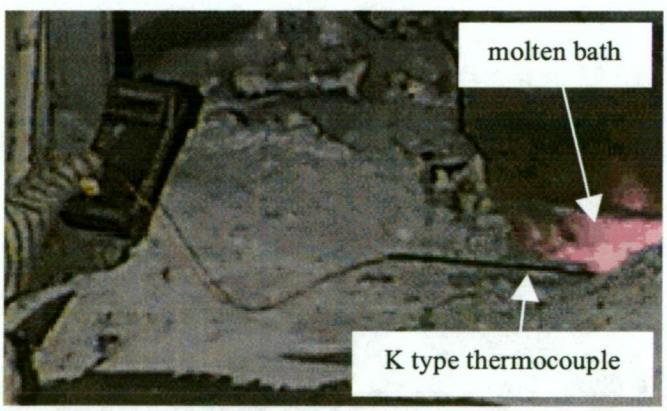


Fig. 4.2.8. Manual Bath Temperature Measurement Technique Highlighting Thermocouple Insertion in Molten Electrolyte

3. Bath height - Bath height specifies the distance between the upper surface of the metal pad and the upper surface of the electrolyte, as shown in Figure 4.2.7. Bath

depths are measured using a mild steel rod, bent in the shape of the letter 'L'. The probe is inserted into the molten electrolyte, a shown in Figure 4.2.9, for approximately 3.0 to 5.0 seconds. The film of bath that is withdrawn with the rod freezes rapidly on cooling, allowing a bath height measurement to be determined. Each completed bath height measurement is recorded and entered into the smelter knowledge base. Bath height is an important parameter as it indicates the volume of electrolyte in the reduction cell. Appropriate bath volume is critical for stable cell operation. Consequently, bath height measurement is a frequent reduction cell task. A study of changing bath height with changing bath chemistry has shown that high bath height generally yields higher electrolyte additions in order to maintain a homogeneous electrolyte solution. Typical values of bath height are in the range of 180.0 to 250.0mm.

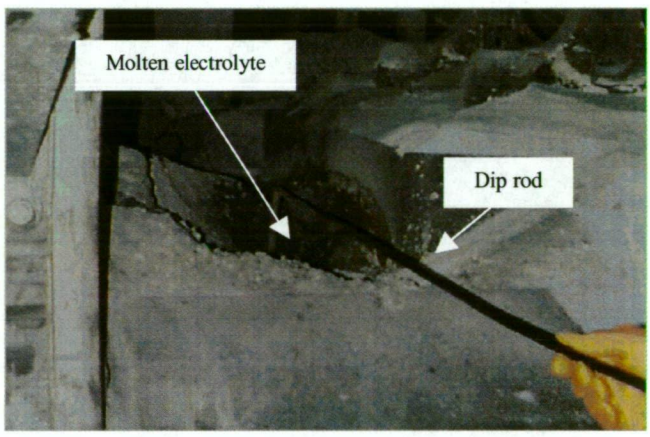


Fig. 4.2.9. Manual Bath Height Measurement Technique Highlighting Dip Rod Insertion in Molten Electrolyte

4. Bath resistivity - Bath resistivity is a measure of how much the bath resists the passage of electrical current. Hence, it is an inverse measure of the electrical conductivity of the electrolyte, which has been shown to vary with changing AlF_3 content. From published data on the electrical conductivity of the Hall-Heroult electrolyte [183,184] it is shown that the electrical conductivity of the bath increases with increasing AlF_3 content. Hence, bath resistivity is low for high AlF_3 content and is an important indicator of percent excess AlF_3 present in the bath chemistry. Bath resistivity is also closely related to bath temperature and bath height. Bath resistivity will be high when the cell has low bath height, high percent excess AlF_3 or low bath temperature [185]. Bath resistivity is frequently calculated using automated

techniques and recorded on the smelter knowledge base. Typical values of bath resistivity are in the range of 4.2 to 5.2ohms/mm.

5. Electromotive force - Electromotive force, or emf, is the voltage required for the chemical reactions which make aluminium production take place, usually in the range of 1.65 to 1.70 volts [186]. A study of the relationship between emf and bath chemistry has shown emf to generally be low for high AlF_3 content, and conversely, high for low AlF_3 content. Hence, emf is a useful network input for this application as it is related to bath chemistry. Emf is frequently calculated using combined automated and manual techniques and recorded on the smelter knowledge base.

6. AlF_3 addition (t-1) - AlF_3 addition is a necessary input to the neural network as it specifies the mass of AlF_3 added to the cell in a previous bath addition, at time t-1. As a result of a high AlF_3 addition, bath temperature should reduce significantly. Consequently, the subsequent addition of AlF_3 to the cell should be significantly lower, attributed to the bath temperature already being low. Therefore, a relationship between the immediate AlF_3 addition and the previous AlF_3 addition exists; the higher the previous AlF_3 addition then lower is the immediate AlF_3 addition. Each AlF_3 addition scheduled to each reduction cell is recorded and entered into the smelter knowledge base. Typical additions of AlF_3 are in the range of 0.0 to 70.0kg.

7. Na_2CO_3 addition (t-1) - Na_2CO_3 addition is a necessary input to the network as it specifies the mass of Na_2CO_3 added to the cell in a previous bath addition, at time t-1. The mass of AlF_3 to add in an addition is inversely proportional to the mass of Na_2CO_3 added in previous additions to the cell. Specifically, the higher the previous addition of Na_2CO_3 then lower is the immediate addition of AlF_3. The mass of Na_2CO_3 to add to a cell varies with bath temperature according to the relationship; the lower the bath temperature the higher is the mass of Na_2CO_3 added. It follows then that for large Na_2CO_3 additions in the last previous addition it is not necessary to add large quantities of AlF_3 in the current addition. Each Na_2CO_3 addition scheduled to each reduction cell is recorded and entered into the smelter knowledge base. Typical additions of Na_2CO_3 are in the range of 0.0 to 70.0kg.

8. Cell power - Power is an important input as it specifies the amount of energy being supplied to the cell for aluminium production. Power is a measure of voltage and current within the cell, given by the following equation [187]:

$$Power = Voltage \text{ (V) x } Current \text{ (Amps)} \quad \text{[Watts]} \tag{4.2.1}$$

All cells in a potline have exactly the same current flowing through them as they are connected in series. Hence, to change the power supplied to a reduction cell it is necessary to change the voltage. This is achieved by changing the anode-to-cathode distance (ACD), which is the distance between the anode bottom surface and the top surface of the metal pad, by lowering or raising the anodes. Within the reduction cell, the power is that given or absorbed by the portion of the circuit across which the voltage is measured, this circuit element also carries a particular current. Voltage changes are partially the result of bath resistivity variation, which consequently varies significantly with bath chemistry. It has been found that high bath resistivity requires additional voltage, consequently cell power increases. Hence, cell power has been included as a network input in this instance as it has some correlation with bath chemistry. Cell power is frequently calculated using automated techniques and recorded on the smelter knowledge base. Typical values of cell power are in the range of 420.0 to 470.0kW.

9. Cell age - Cell age is an important consideration when scheduling AlF_3 and Na_2CO_3 additions to the electrolyte as the amount to add varies with cell age. In considering this it is important to note that sodium absorbs into the carbon cathode lining and carbon lining materials of the cell during cell operation. Initially the rate of absorption is very high such that up to 400.0kg of Na_2CO_3 may be required to be added to the cell in the first week of cell operation [39]. The soda absorption rate then decreases initially very rapidly ceasing when the cells have reached an age of approximately 800 to 1,000 days. The amount of AlF_3 required becomes constant after about 800 to 1,000 days once the sodium absorption has virtually ceased or reached a constant low level [39]. Over the first 800 to 1,000 days of a reduction cells life, Na_2CO_3 additions to the cell range from approximately 400.0kg per cell per week on average for 0 to 100 days of operation, through to 250.0kg of AlF_3 for 800 to 1,000 days of operation. Hence, cell age is an important variable to include as a

network input because it has significant influence over the amount of AlF_3 and Na_2CO_3 to add to the cell. The age of each reduction cell is automatically recorded each day to the smelter knowledge base. As a reduction cell does not exceed a life span of more than 2,500 days, then cell age is in the range of 0 to 2,500 days.

10. Fluoride content of alumina - Frequent additions of alumina are made to the industrial reduction cell to compensate for alumina consumption during aluminium production. Hence, the alumina, which contains small levels of fluoride (F), generally in the range of 1.4 to 2.0% by weight, is a source of F addition to the cell. Consequently, the addition of F to the cell through alumina feeding must be considered in the calculation of excess AlF_3 and Na_2CO_3 to add. The higher the F content of the alumina then lower is the amount of excess AlF_3 to add, or conversely, higher is the amount of Na_2CO_3 to add. The F content of the alumina is determined daily using laboratory analysis and recorded on the smelter knowledge base.

11. Sodium content of alumina - Similarly, alumina contains traces of sodium (Na), generally in the range of 0.5 to 0.7 % by weight. Hence, alumina feeding is a source of Na additions to the reduction cell. Na reacts with AlF_3 to form cryolite, reducing the excess AlF_3 content of the bath. As Na increases the required excess AlF_3 to add, or conversely, decreases the required Na_2CO_3 to add, then it must be included as a network input. The Na content of the alumina is determined daily using laboratory analysis and recorded on the smelter knowledge base.

12. Temperature reference - Temperature reference is necessary to reference the network inputs of bath temperature and bath height, with respect to the network outputs. Bath temperature and bath height are measured simultaneously every 48.0 hours, while AlF_3 and Na_2CO_3 additions are made every 24.0 hours. Hence, the last previous bath temperature and height measurement can potentially be made up to a maximum of 36.0 hours prior to an electrolyte addition and a minimum of 0.0 hours prior to an electrolyte addition. As a means of specifying when the last available bath temperature and height measurements were taken it is useful to use temperature reference as a neural network input.

Output 1. AlF_3 addition (t) - The specific objective of the phenomenological model of the Hall-Heroult process in this instance is to provide improved control of bath temperature. As a means of achieving this, control of the electrolyte chemistry is paramount. Hence, one output of the network is the AlF_3 addition mass, at time t. AlF_3 addition mass as an output of the network is the amount of AlF_3 to add to maintain or return the bath temperature to the desired target of 965.0°C.

Output 2. Na_2CO_3 addition (t) - Further, the Na_2CO_3 addition mass is also a necessary network output as it is a critical element of bath chemistry control and hence, bath temperature control. Similarly, Na_2CO_3 addition mass as an output of the network is the amount of Na_2CO_3 to add at time t to maintain or return the bath temperature to the desired target of 965.0°C.

4.3 APPLICATION OF NEURAL NETWORKS TO PREDICT CELL FAILURE

The application being scoped here involves the use of neural networks to predict which reduction cells will fail in a given period and therefore should be removed from production. It is useful to note that reduction cell failure is a term used to describe a cell that can no longer continue to remain in production due to substantial deterioration of the reduction cell materials. While it will be shown that there are many factors influencing reduction cell life, it is important to note that there exists substantial economic benefit in having a technique for predicting the time that a particular cell will fail. A neural network is applicable to this application as there are many parameters within the reduction cell that are currently measured that are possible indicators of a cell failure occurrence. These parameters can be retrieved from the smelter knowledge base to develop training and test data for a neural network. However, before detailing the particularities of this application it is useful to provide some detail on cell failure mechanisms in reduction cells.

4.3.1 Cell Failure Mechanisms in Aluminium Reduction Cells

There are many identifiable causes that contribute to the ultimate failure of an aluminium reduction cell. They can be broadly classified within the bounds of cell design, materials of construction, method of construction and preheating and cell operational history [188]. Ideally, a reduction cell cathode should last as long as it

takes the natural erosive and corrosive forces to wear the carbon cathode lining evenly down to the level of the current collector bars [189]. However, in practice this does not occur and reduction cells fail in a significantly shorter period of time. One of the main factors influencing the cost of metal production and subsequently, the profitability of an aluminium electrolysis cell, is the life of the cell lining [190]. In particular, the longer the life of the cell lining materials then longer is the life of the reduction cell and consequently, lower are the associated costs of aluminium production. In an attempt to maximise profit associated with aluminium production, smelters constantly strive to improve cell design and the quality of cell lining materials as a strategy to increase cell life. In addition, careful attention is given to the construction and preheating of the cell together with improved operational strategies to increase cell life. However, when cell failure occurs it is usually attributed to failure of the carbon cathode and is typically attributed to cathode delamination, longitudinal cracking, ramming paste shrinkage, pothole formation, sidewall tap-out or taphole formation. A brief note on each of these cell failure mechanisms is useful to facilitate an understanding of this particular application.

Cathode Delamination - Cathode delamination refers to the development of substantial cracks in the carbon cathode, as shown in Figure 4.3.1, typically as a result of significant thermal stresses in the reduction cell and poor quality carbon cathodes. Consequently, molten aluminium enters these developed cracks and travels to the steel collector bar where it rapidly erodes the collector bar and creates a pathway to flow from the reduction cell to the surroundings. As a result the affected reduction cell must be removed from operation.

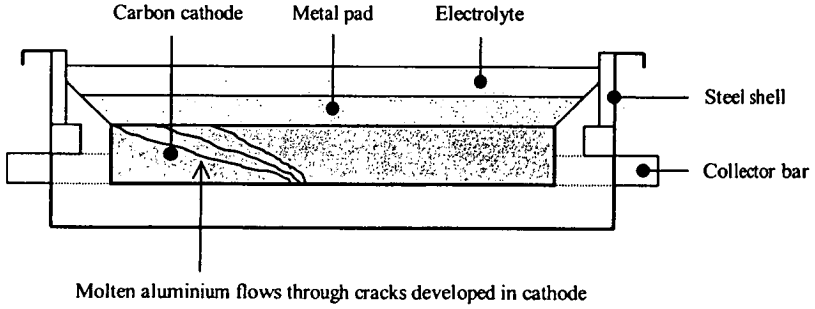


Fig. 4.3.1. Cross-Sectional Illustration of Cathode Delamination in an Aluminium Reduction Cell

Longitudinal Cracking - The steel collector bar in reduction cell cathodes is normally sealed to the bottom block by cast iron, carbonaceous ramming paste, glues and cements. Since the thermal expansion coefficient of the collector bar can be several times that of carbon, severe thermomechanical stresses can develop when the assembly is heated [189]. Such stresses can lead to severe crack formation that may completely destroy the cathode, as shown in Figure 4.3.2. As a result molten aluminium enters the cracks and eventually reaches the collector bar, which it rapidly erodes, resulting in molten aluminium flowing from the reduction cell to the surroundings.

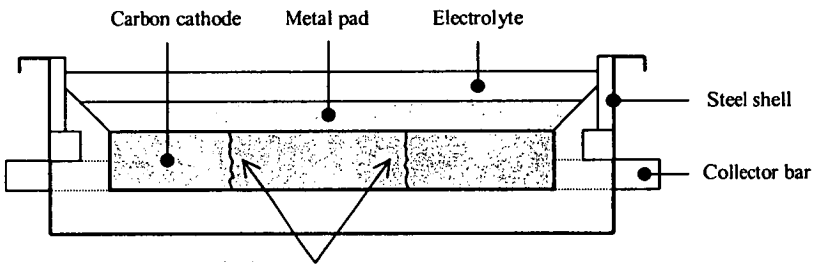


Fig. 4.3.2. Cross-Sectional Illustration of Longitudinal Cracking in an Aluminium Reduction Cell

Ramming Paste Shrinkage - Carbonaceous ramming paste is used during construction of the reduction cell to seal and fill any gaps that may be present between adjacent lining materials. However, while baked carbon materials display a positive coefficient of thermal expansion at all temperatures they are likely to experience in the reduction cell, carbonaceous ramming pastes will normally shrink within a certain temperature range [191]. High ramming paste shrinkage can result in a vertical crack opening between lining materials and hence, be detrimental to reduction cell life, as molten aluminium follows this route to eventually flow from the cell to the surroundings, as shown in Figure 4.3.3. As a consequence, the cell must be disconnected from the power supply and removed from operation.

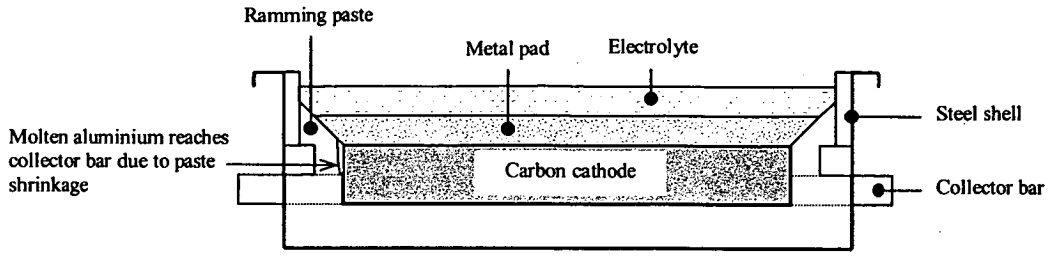


Fig. 4.3.3. Cross-Sectional Illustration of Ramming Paste Shrinkage in an Aluminium Reduction Cell

Pothole Formation - Under certain circumstances, wear of the carbon cathode can be very rapid and the cell fail due to very localised erosion of the carbon lining. This type of erosion is termed pothole formation since the fatal failure area often has the form of a narrow depression or hole through the carbon lining [189]. The initial phase of a pothole formation is probably a small stationary area of local high current density at the cathode carbon surface, most likely due to a local weakness at the surface that allows a minor amount of molten aluminium to penetrate. The higher electrical conductivity of the metal compared to carbon will increase current density and give rise to a localised magnetodynamic disturbance in the metal [189]. This yields localised erosive action that accelerates as the bottom of the pothole formation moves towards the current collector bar, as shown in Figure 4.3.4. The cell may then fail in a very short period once aluminium contacts the collector bar, as molten aluminium quickly erodes the collector bar and eventually flows from the cell to the surroundings.

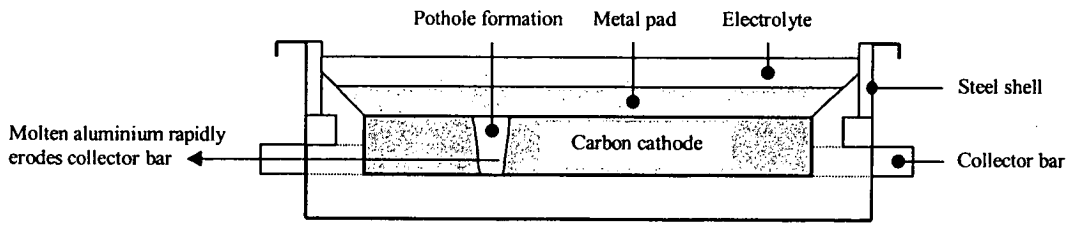


Fig. 4.3.4. Cross-Sectional Illustration of Pothole Formation in an Aluminium Reduction Cell

Sidewall Tap-out - Sidewall tap-out refers to the procedure whereby molten aluminium flows from the cell through the upper section of the steel shell, as shown in Figure 4.3.5. This occurs as a result of molten electrolyte contacting the lining

materials in the upper section of the cell. Typically, a protective layer of ledge forms on the lining materials in this section to prevent molten electrolyte from contacting the lining. However, temperature fluctuations within the cell and inadequate insulation requirements can result in the loss of this protective side ledge. Hence, molten electrolyte rapidly erodes the lining materials and ultimately results in cell failure.

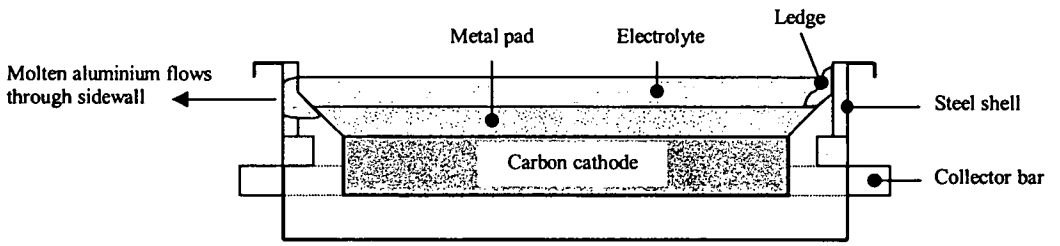


Fig. 4.3.5. Cross-Sectional Illustration of Sidewall Tap-out in an Aluminium Reduction Cell

Taphole Formation - During normal operation the carbon surface of the cathode experiences slow and steady wear during the life of the cell. However, periodic metal removal, or 'tapping', can increase carbon surface wear locally as tapping typically has to be completed at a fixed location and there is no bottom ledge to protect the carbon surface. Moreover, the increased flow of metal at this localised point during tapping substantially increases erosion and a taphole is formed in the cathode as a consequence. Eventually the bottom of the taphole reaches the collector bar, as shown in Figure 4.3.6.

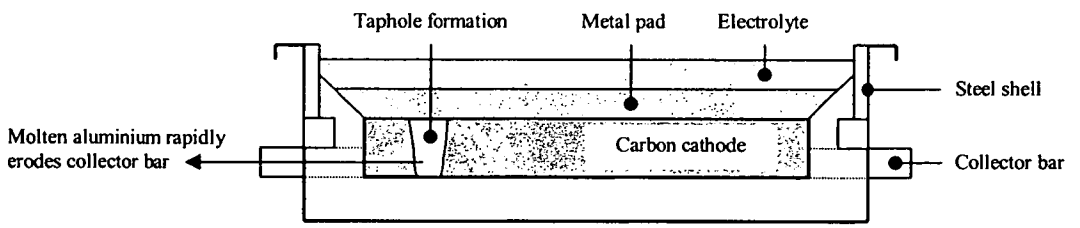


Fig. 4.3.6. Cross-Sectional Illustration of Taphole Formation in an Aluminium Reduction Cell

Consequently, molten aluminium rapidly erodes the collector bar and flows from the reduction cell to the surroundings, causing the cell to be removed from production.

Taphole formation differs from pothole formation in that taphole formation is consistently restricted to the same position in the reduction cell, the point where metal tapping occurs and is attributed to a particular operational procedure. Pothole formation, on the other hand, occurs randomly on the surface area of the carbon cathode and it is not clearly understood what mechanisms cause this particular failure mode.

A study of percentage contribution of these cell failure mechanisms towards the number of total cell failures per annum has shown that pothole formation is responsible for approximately one-half of all cell failures, followed by taphole formation accounting for approximately one-quarter of all cell failures. Further, cathode delamination and longitudinal cracking of the cathode are substantial contributors to overall cell failure, 15.0 and 8.0%, respectively, while sidewall tap-out and paste shrinkage are less significant, 3.0 and 1.0%, respectively.

4.3.2 Existing Cell Failure Prediction Technique at CABBL

The existing cell failure prediction technique at CABBL was developed with the focus that cell failure prediction is a significant contributor to materials ordering, yearly planning and quarterly forecasting. Consequently, the existing failure prediction methodology is concerned with the number of cells that will fail in a specified period, rather than a prediction of the particular cells that will fail in a specified period. Moreover, while the existing failure prediction methodology assists such tasks as materials ordering and quarterly cost forecasting it does not assist in determining those reduction cells that are at high risk of failure. In particular, the existing failure prediction methodology used at CABBL incorporates fitting a Weibull distribution to a Kaplan-Meier [192] estimate of the survival function of all existing operating reduction cells. This technique is used to estimate how many cells will fail in a given period. On the other hand, the decision to remove a particular reduction cell from production prior to its failure is aided by a metal analysis technique. Specifically, the decision on whether to remove a reduction cell from operation that is suspected to fail in the short term and consequently, tap-out involves analysis of molten aluminium from each reduction cell for impurity quantities. In particular, a spectrographic analysis of the molten aluminium, which is routinely completed at a frequency of 96.0 hours per reduction cell, highlights, among other

elements, the iron, Fe, and silicon, Si, content of the aluminium. If the Fe content of the molten aluminium is particularly high it is possible that the cathode has eroded significantly to allow molten aluminium to erode the steel collector bar and hence, the Fe from the collector bar is present in the molten aluminium. If the collector bar is being eroded the Fe content of the molten aluminium will continue to rapidly increase and ultimately a tap-out will occur as the consequence of cell failure. However, it is possible for Fe to enter the molten aluminium as a result of contacting the steel rods connecting the anodes to the electricity supply. Fe in the molten aluminium from this source, however, while initially high, will dilute during further aluminium production to an acceptable level without consequence, as shown in Figure 4.3.7.

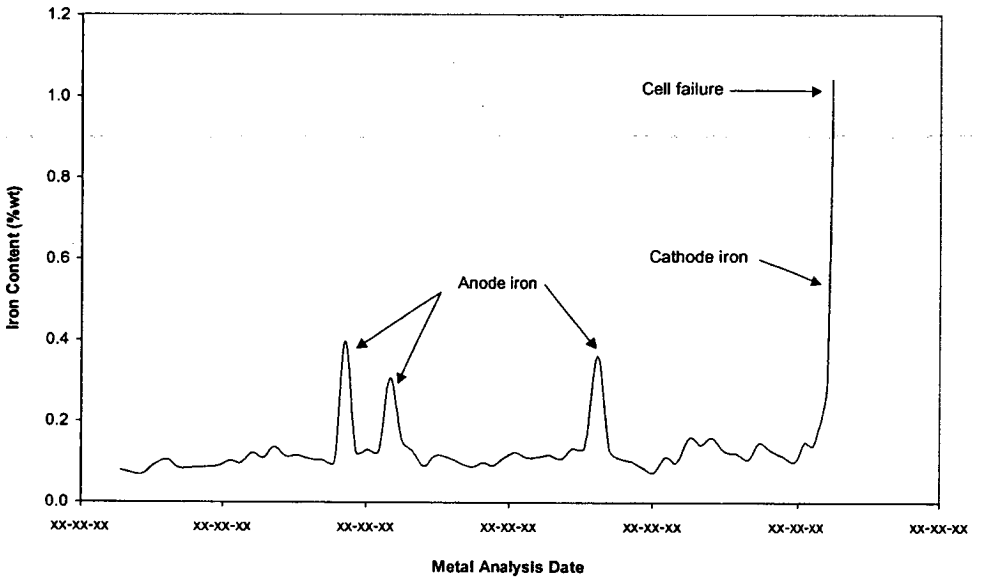


Fig. 4.3.7. Illustration of Anode and Cathode Iron in Molten Aluminium during a Period of Cell Life

Thus it becomes important to identify whether the source of Fe in the molten aluminium is anode or cathode related. While there is no known reliable technique for this identification, a useful methodology is to increase the frequency of the spectrographic analysis of the molten aluminium to observe whether the high Fe content is increasing or diluting with increasing time. If the Fe content is found to increase with time the cell is removed from production before failure occurs and is referred to as a 'cut-out'. Similarly, if the Si content of the molten aluminium is

particularly high it is attributed to sidewall lining material deterioration and the reduction cell is consequently cut-out before a sidewall failure occurs. However, because cell failure occurs rapidly in most instances and because a spectrographic analysis of the molten aluminium is only completed every 96.0 hours per cell under normal production conditions, it is common that a reduction cell can fail while no indication was given by the most recent spectrographic analysis. Due to this phenomenon, approximately 50.0% of all reduction cells removed at CABBL are the result of a failure, while the remaining are cut-out. On the other hand, if a cell is found to have a high Fe content, more frequent metal analysis is completed to monitor the behaviour of Fe content in the aluminium. While this is necessary to avoid cell failure occurrences, substantial labour is required to increase the frequency of metal sampling. Hence, it is important that a cell failure prediction strategy be introduced that can reliably forecast cell failure time accurately, maximising cell life while eliminating failure occurrences. Neural networks are applied in this instance as an intelligent decision making apparatus. The economic benefit of such a strategy is highlighted in the following justification.

4.3.3 Rationale for Reduction Cell Failure Prediction Using Neural Network Modelling

It is important to note that the occurrence of molten aluminium flowing from the reduction cell to the surroundings as a result of cathode or sidewall failure, attributed to one of the listed cell failure mechanisms, is referred to as a 'tap-out'. Following a tap-out, the reduction cell must have the supply of power disconnected and be removed from production as a consequence. If left in production, molten aluminium would continue to flow from the reduction cell, as shown in Figure 4.3.8, causing substantial damage to the cell surroundings. In addition, molten aluminium that flows from the cell entrains contaminants as it solidifies and hence, can only be sold as scrap at a significantly lower rate than the purer metal produced under normal operating circumstances. Hence, it is critical that molten aluminium not be allowed to flow from the reduction cell as a result of cell failure. Therefore, it is necessary to identify cell failure prior to its occurrence and remove the reduction cell to avoid metal spillage.

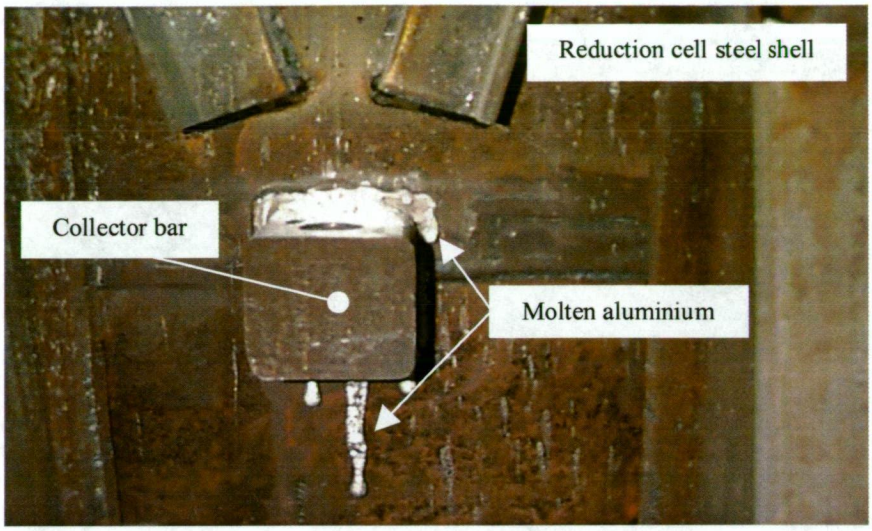


Fig. 4.3.8. Illustration of Reduction Cell Tap-Out Highlighting Aluminium Loss via Cathode and Collector Bar

Hence, the occurrence of a tap-out leads to lost production costs, material damage and labour costs associated with maintenance and necessary repair work. The frequency of tap-outs at CABBL is at a rate of approximately forty-two per year, yielding substantial economic cost per annum. There is potential to significantly reduce the cost associated with cell failure through implementation of neural networks to accurately predict a cell failure occurrence, allowing the cell to be removed immediately prior to a tap-out occurrence. However, it is also important to note that there is an economic cost associated with removing a reduction cell from operation prior to cell failure. In particular, removing a reduction cell that would not have failed, but was falsely predicted to fail, from production results in an economic loss. Specifically, if a reduction cell is predicted to fail, but indeed would not if left in production, it is removed, resulting in lost metal production from that cell. Moreover, removing a cell from production substantially prior to failure results in unnecessary replacement costs, such as labour, materials and equipment. Hence, it is important that an optimum cell life be achieved, which requires removing the reduction cell from operation immediately prior to a tap-out occurring, achieving the maximum cell life possible without allowing a tap-out to occur. Therefore, the prediction strategy implemented must be capable of identifying a tap-out as close as possible to the occurrence without allowing a tap-out to occur.

In addition, it is useful to note that while the contents of the failed cell are removed and discarded, the steel shell is refurbished and new lining materials are placed in the refurbished shell. A sufficient number of these new cells are maintained, ready to be placed in production to replace removed failed cells. The time lapse between a failed cell being disconnected from the power supply and a new cell being installed and connected is referred to as 'cell-turn-around-time' and is typically in the range of 20.0 to 25.0 hours at CABBL. It is important from an economic viewpoint that a reduction cell site is left empty only for a minimum period of time, as the amount of aluminium produced is directly proportional to the number of cells operating. Hence, it is important that failed cells be immediately replaced with new cells and it is the endeavour of process improvements at CABBL to minimise cell-turn-around-time. However, a tap-out generally causes significant damage to the cell surroundings, giving rise to an increased cell-turn-around-time, as substantial maintenance is generally required to repair the cell surroundings to an acceptable state. In particular, a tap-out typically adds approximately 15.0 to 20.0 hours to cell-turn-around-time. Hence, it is desirable to eliminate tap-outs, removing the reduction cell from production immediately prior to a tap-out occurrence.

4.3.4 Specification of Neural Network Inputs for Cell Failure Prediction Application

Similar to the previous application, the parameters used as network inputs for this particular application are measured parameters of the Hall-Heroult process. Further, the parameters specified here include any variables that are considered to provide at least some indication of cell failure time. While the network inputs selected here may not contribute to the network prediction, it has been noted that is not until neural network modelling is evaluated that this conclusion can be drawn. Hence, all parameters that may have an influence on the prediction of a cell failure are included and eliminated from the model if shown not to contribute. An illustration of the neural network model used for this particular application is shown in Figure 4.3.9, highlighting the twenty-four potential input parameters and the single output variable. It is useful to note here that while an output value of either 0.0 or 1.0 is required, the actual output value of the neural network can be any value in the bounded range of the activation function used in the output layer. For example, the output of a neural network using the sigmoidal activation function in the output layer

can be any value in the bounded range 0.0 to 1.0. Hence, it is necessary to use an interpreter on the neural network to convert the output value to either 0.0 or 1.0. In this instance, because the activation functions used in the output layer of the applied neural networks have a bounded range of 0.0 to 1.0, the interpreter performs a rounding procedure. Specifically, the output of the interpreter, int_{out}, is calculated using the following:

$$int_{out} = \begin{cases} 0, \text{if neural network output} < 0.5 \\ 1, \text{if neural network output} \geq 0.5 \end{cases} \tag{4.3.1}$$

Therefore, any output value of the neural network that is less-than 0.5 is rounded to 0.0, while a neural network output value greater-than-or-equal-to 0.5 is rounded to 1.0.

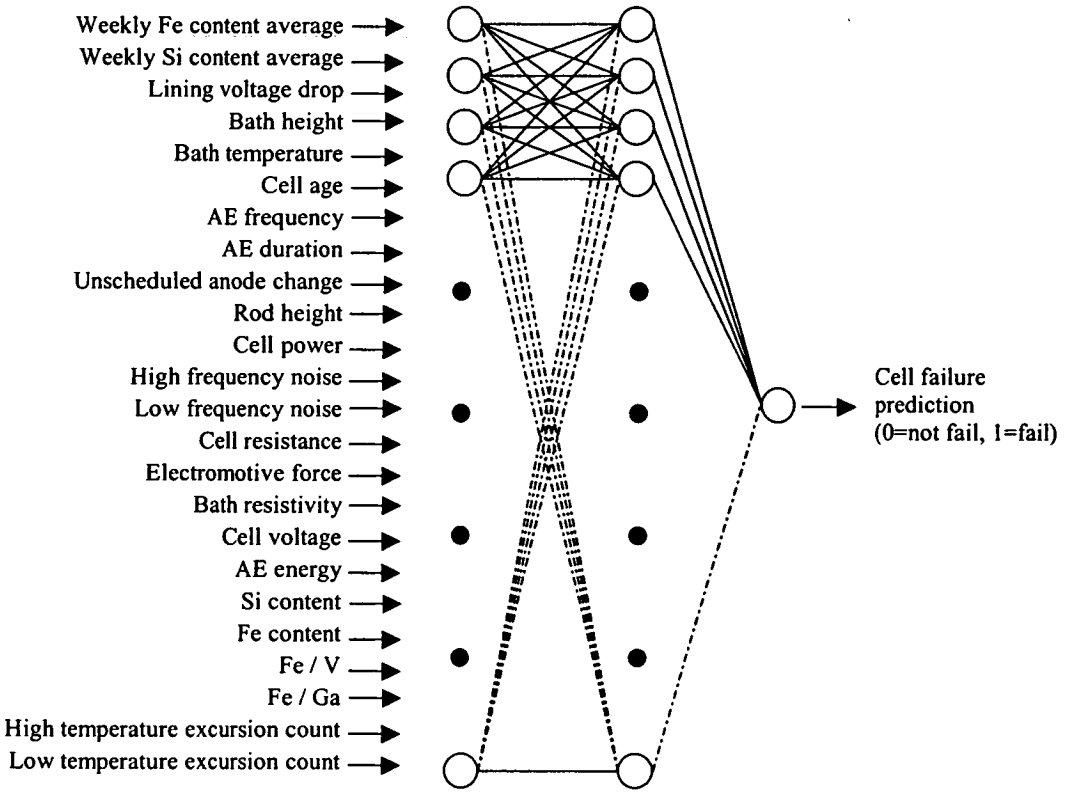


Fig. 4.3.9. Illustration of Neural Network Model for Cell Failure Prediction Application

The following section gives a brief description of each process parameter used as a neural network input for this application and provides a justification for each

parameters inclusion in the model. It is useful to note that while some of the process parameters used as input variables for this application have been previously discussed, they are reiterated here to highlight their justification for inclusion in the neural network for cell failure prediction. To aid the discussion, Figure 4.3.10 highlights the association of the input variables with the reduction cell.

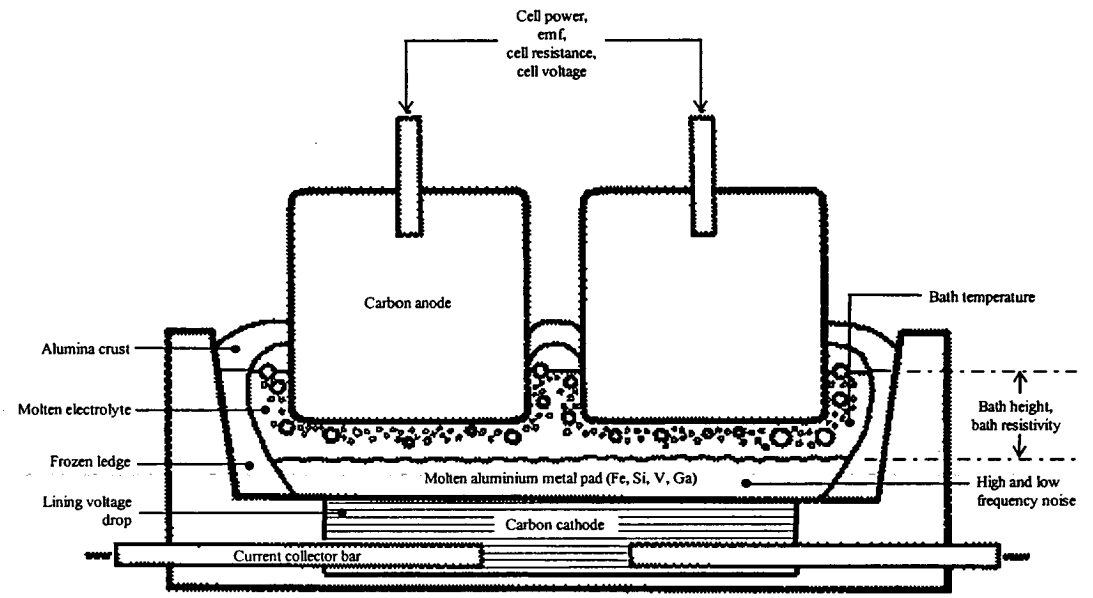


Fig. 4.3.10. Cross-Section of Reduction Cell Indicating Process Parameters used as Neural Network Inputs for Cell Failure Prediction Application

1. Weekly Fe content average - Fe content refers to the iron content of the molten aluminium. Molten aluminium samples are collected from each reduction cell using a heavy duty ladle, as shown in Figure 4.3.11(a), and poured into a mould to form a solidified metal sample, as shown in Figure 4.3.11(b). Subsequently, the metal sample is analysed by spectrographic analysis. This procedure allows the Fe content of the aluminium in each reduction cell to be determined and recorded on the smelter knowledge base. It has been shown that Fe content is a useful indicator of cell failure, as high Fe content can be an indication of collector bar erosion, which is detrimental to cell life. The weekly average Fe content is used here to provide some indication of whether the Fe content is due to an anode or cathode source. Because anode Fe dilutes quickly while cathode Fe increases in the molten aluminium with increasing time, the weekly Fe content average will be low for anode Fe but high for cathode Fe.

Typical values for weekly Fe content average are in the range of 0.01 to 1.50% by weight.

(a) (b)

Fig. 4.3.11. Collection of Metal Samples from Reduction Cell for Spectrographic Analysis. (a) Molten Aluminium Collection, and (b) Solidified Metal Sample

2. Weekly Si content average - Si content refers to the silicon content of the molten aluminium, determined using spectrographic analysis of aluminium samples collected from the reduction cells. The Si content of the aluminium in each reduction cell is frequently determined and recorded on the smelter knowledge base. Si content is a useful network input as it highlights erosion of the high Si content sidewall lining materials used in the reduction cell, which is detrimental to cell life. As Si can enter the cell through alumina and carbon contaminants, which rapidly dilute in the molten aluminium, a weekly Si content average is used here to identify Si from sidewall lining materials. Si from the sidewall will continue to increase in the molten aluminium with increasing time and therefore increase the weekly Si content average, while Si from other sources will dilute and have minor significance on the weekly average value. Typical values for weekly Si content average are in the range of 0.01 to 0.30% by weight.

3. Lining voltage drop - The carbon cathode, which is a conductor of electricity within the reduction cell, deteriorates during the production life of the cell. Consequently, the electrical conductivity of the carbon cathode also deteriorates. A voltage drop measured across the cathode is used to identify the electrical resistance

of the carbon material, which will increase with increasing cell age. A cathode that is substantially deteriorated and close to failure will yield poor conductivity. Hence, lining voltage drop is a useful indicator of cell failure. Each completed lining voltage drop for a reduction cell is recorded on the smelter knowledge base. Typical values of lining drop voltage are in the range 250.0 to 450.0mV.

4. Bath height - Bath height is included as a network input at this stage of the modelling process as it may contribute towards the network prediction accuracy. While there are no observed trends between bath height and cell failure it is interesting to include this parameter here and discard it during network modelling if found to be unnecessary. The technique used for manual bath height measurement is noted in the process parameter discussion for the previous application. Bath height is typically in the range of 180.0 to 250.0mm.

5. Bath temperature - Bath temperature is useful as a network input in this instance. It is typically shown through historical data from the smelter knowledge base that the average bath temperature of a reduction cell increases slightly with increasing cell age. Hence, bath temperature may provide some useful data to the neural network for predicting cell failure. The technique used for manual bath temperature measurement is noted in the process parameter discussion for the previous application. Typical values of bath temperature are in the range of 935.0 to 995.0°C.

6. Cell age - Cell age is a necessary network input for this particular application. The days remaining until cell failure occurs is highly correlated with the number of days a cell has been in production. Specifically, the higher the cell age then higher is the probability that a cell will fail. Cell age is in the range of 0 to 2,500 days.

7. AE frequency - AE, or anode effect, frequency refers to the number of AE's a cell has during a given time period. AE is a term used to describe a sudden rise in cell voltage from a nominal 4.0 to 5.0 volts to a high level, usually 25.0 to 35.0 volts. An AE occurs when the electrolyte becomes depleted in alumina, and results in increased energy consumption, reduced metal production and overheating of the cell [2]. During an anode effect fluoro-carbon compounds form on the anode block surface producing an insulating film, which only allows current to flow by arcing, which is

the jumping of electrical charge across an air gap between two conductors to produce sparking and noise. Specific process control equipment is used to identify an AE at CABBL and administer corrective procedures to terminate the AE within a short time of its initialisation. However, due to overheating of the cell during the AE occurrence it is suspected that AE's have a correlation with cell failure as thermal cycling deteriorates the reduction cell lining and cathode materials. Hence, AE frequency is included as an input for this application, used to identify the number of AE's a particular cell has been subjected to. The frequency of an AE occurrence is automatically recorded to the smelter knowledge base. Typical AE frequency is at a rate of 0.05 to 3.00 per day per reduction cell.

8. AE duration - In addition to AE frequency, AE duration is used to specify the length of the AE occurrence. The duration of an AE can be as short as 20.0 seconds in some instances, or as long as 5.0 minutes in extreme circumstances. Moreover, the damage to the reduction cell occurring as a result of an AE is proportional to the AE duration, the longer the AE event then higher is the deterioration of the reduction cell lining and cathode materials. Hence, it is important to have a measure of AE duration as a network input for this application as it is correlated with cell failure. The duration of an AE occurrence is automatically recorded to the smelter knowledge base.

9. Unscheduled anode change - While anodes are consumed at a consistent rate and replaced routinely during aluminium production, an unscheduled anode change is sometimes required if particular anode problems occur, causing an anode to be prematurely consumed. An unscheduled anode change causes significant disturbances in the cell operating conditions, including thermal cycling and therefore may be a possible contributor to cell failure. The value of this input represents the number of unscheduled anode changes that have occurred in the life of the reduction cell. Whenever an unscheduled anode change is completed on a reduction cell the incident is recorded on the smelter knowledge base. Typical values for unscheduled anode change are in the range of 0 to 40 during reduction cell life.

10. Rod height - The steel bar connecting an anode to the electricity supply and used to connect an anode to the reduction cell is referred to as a 'rod', as shown in Figure

4.3.12. Each anode has its own rod, which is attached to a fixed beam on the reduction cell to ensure all anodes maintain a uniform profile relative to the cathode surface. The position of a reference mark on each rod relative to the fixed beam is termed the 'rod height'. Due to bowing of the cathode with increasing production life and in order to maintain a uniform profile between the anode and cathode surface it is necessary during the life of the cell to sequentially increase the anode rod height. Hence, rod height is a useful network input as it has some relationship with cathode age, which is related to cell failure. Rod height is frequently measured for each reduction cell and recorded on the smelter knowledge base. Typical values of rod height are in the range of 200.0 to 400.0mm.

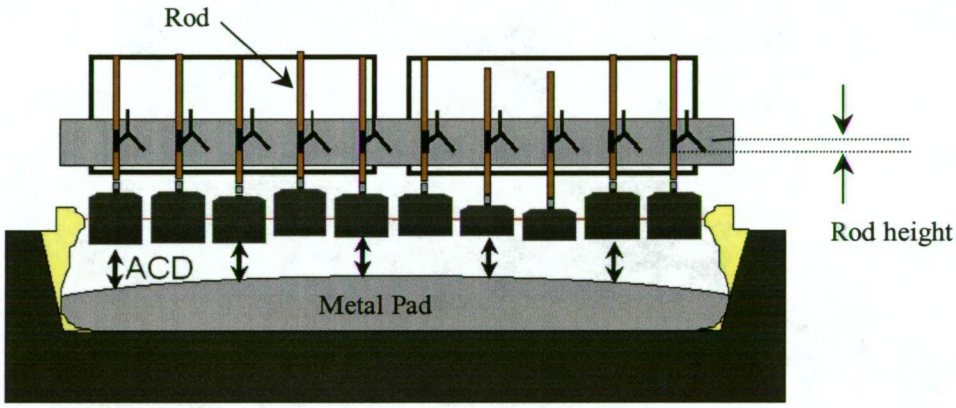


Fig. 4.3.12. Illustration of Anode Rod and Rod Height Relative to the Reduction Cell

11. Cell power - Cell power is an important network input in this instance as it specifies the amount of energy being supplied to the cell for aluminium production. A typical trend of increasing power consumption with increasing cell age is observed from historical data from the knowledge base. Hence, cell power may be a significant contributor to the accuracy of the neural network for predicting cell failure. Automated data acquisition is used to frequently determine cell power for each reduction cell and record the appropriate value to the smelter knowledge base. Typical cell power values are in the range of 420.0 to 470.0kW.

12. High frequency noise - High frequency noise is a useful measure of instability within a reduction cell. Moreover, significantly high values of high frequency noise are observed close to cell failure. Hence, high frequency noise is useful as a network input in this instance. The high frequency noise of a reduction cell is measured using

automated techniques and recorded on the smelter knowledge base. High frequency noise values are typically in the range of 0.02 to 0.05μohms/s.

13. Low frequency noise - Similarly, low frequency noise is also a measure of instability in the reduction cell. In addition, as cell age increases, low frequency noise has been shown to increase. Moreover, reduction cells close to failure exhibit substantially high values of low frequency noise. Hence, this parameter is useful as a network input for this application. The low frequency noise of a reduction cell is measured using automated techniques and recorded on the smelter knowledge base. Low frequency noise values are typically in the range of 0.05 to 0.25μohms/s.

14. Cell resistance - Cell resistance is a measure of the ability of electricity to flow through the reduction cell. High cell resistance impedes electricity flow, decreasing the current efficiency of a reduction cell. Typical trends of cell resistance have shown this parameter to increase as reduction cell age increases. Hence, older cells exhibit higher values of cell resistance, due mostly to cathode deterioration. Therefore, cell resistance is a useful network input for this particular application. Cell resistance is measured automatically and recorded on the smelter knowledge base. Typical values of cell resistance are in the range of 25.0 to 35.0μohms.

15. Electromotive force - It has been stated that electromotive force, or emf, is the voltage required for the chemical reactions which make aluminium production take place. In particular, as reduction cell age increases, emf is also shown to increase, due mostly to a significant decrease in the current efficiency of a cell as it deteriorates. Emf is frequently calculated using combined automated and manual techniques and recorded to the smelter knowledge base.

16. Bath resistivity - While there are no witnessed trends between bath resistivity and cell failure it is interesting to include this parameter at this stage of the modelling process to observe if there is indeed a correlation. Neural network modelling will highlight the contribution this particular parameter has in the model. If it is shown that there is no contribution from bath resistivity it can be removed from the network model. Bath resistivity is frequently calculated using automated techniques and

recorded to the smelter knowledge base. Typical values of bath resistivity are in the range of 4.2 to 5.2ohms/mm.

17. Cell voltage - Similar to cell power, cell voltage specifies the supply of electricity to a reduction cell. A typical trend exhibited by reduction cells is that cell voltage increases during increasing production life, due to higher resistance within the reduction cell attributed to material deterioration. Hence, cell voltage has some relationship with cell age and consequently cell failure. Cell voltage is frequently calculated using automated techniques and recorded to the smelter knowledge base. Typical values of cell voltage are in the range of 3.5 to 5.0V.

18. AE energy - In addition to AE frequency and duration, a further measure of this disruptive event is AE energy, which is the amount of energy consumed by the AE occurrence. AE energy is calculated as the sum of AE duration multiplied by the power consumed during the AE event, specified in units of joules (J). Hence, AE energy is a useful value for identifying the severity of AE occurrences during a given period. The deterioration caused to the reduction cell as a result of the AE event is proportional to AE energy. The energy consumed during an AE occurrence is automatically recorded to the smelter knowledge base. Typical values of AE energy are in the range of 1.0 to 20.0kJ per day per reduction cell

19. Si content - Si content in this instance specifies the most recent Si content of the molten aluminium, determined using a single metal sample and spectrographic analysis. It is included as a network input for this application as it may be a relevant indicator of cell failure, in addition to the previously specified weekly Si content average. It has been stated that Si content of the molten aluminium is a useful indicator of sidewall lining material damage and consequently, cell failure. Typical values for Si content are in the range of 0.01 to 0.05% by weight.

20. Fe content - Similarly, Fe content in this instance specifies the most recent Fe content of the molten aluminium, determined using a single metal sample and spectrographic analysis. It is included as a network input in this instance as it may be a relevant indicator of cathode deterioration and therefore cell failure, in addition to

the previously specified weekly Fe content average. Typical values for Fe content are in the range of 0.01 to 1.50% by weight.

21. Fe/V - The ratio of Fe to V, vanadium, which is an impurity found in molten aluminium as a result of the raw materials used during smelting, is useful to identify the source of Fe in the metal sample. Recent analysis [193] in operating cells with different forms of iron, metallic and oxide, have shown that recovery of V varies with the form and method of Fe introduction. It is shown that iron oxide, FeO, or more commonly, rust, which is introduced to the cell through anode rods, yields an increase in V in molten aluminium, while metallic Fe, from the cathode collector bar, has no effect on V content. Hence, it follows that the Fe to V ratio will remain small and approximately constant for increasing Fe content if Fe is from an anode source, introduced as FeO. On the other hand, if Fe is from a cathode source, which ultimately results in cell failure, the Fe to V ratio will increase with increasing Fe content. Moreover, a high Fe to V ratio indicates cathode Fe while a low ratio indicates anode Fe. Therefore, the Fe to V ratio is useful to identify the source of Fe in the molten aluminium and thus a useful neural network input for this application. The content of V in aluminium is determined using a metal sample and subsequent spectrographic analysis. Typical values for the Fe to V ratio are in the range 3.0 to 85.0.

22. Fe/Ga - In addition, work completed by Cole *et al* [193] has shown that Ga, gallium, behaves in a similar manner to V when Fe is introduced to the reduction cell in metallic and oxide forms. It is shown that increases in Fe content that can be related to FeO entering the metal appear to cause an increased recovery of Ga from raw materials, while metallic Fe has no effect on the Ga content of molten aluminium. The content of Ga in aluminium is determined using a metal sample and subsequent spectrographic analysis. Typical values for the Fe to Ga ratio are in the range 3.0 to 100.0.

23. High temperature excursion count - The value of this input specifies the number of instances a reduction cell has had an electrolyte temperature in excess of 975.0°C, which is considered to be the critical value for which higher temperatures result in deterioration of the reduction cell lining materials. This input is expected to

contribute to network predictions, as high electrolyte temperature is a possible contributor to cell failure. The number of times a reduction cell experiences an electrolyte temperature greater than 975.0°C during its production life is typically in the range of 15 to 100. A high temperature excursion count for a reduction cell is completed by summing the number of bath temperatures on the smelter knowledge base that are above 975.0°C.

24. Low temperature excursion count - Conversely, the value of this input specifies the number of instances a reduction cell has had an electrolyte temperature below 950.0°C, which is considered to be the critical value for which lower temperatures result in deterioration of the reduction cell lining materials. This input is expected to add value to the network prediction for this application as low electrolyte temperature is a possible contributor to cell failure. The number of times a reduction cell experiences an electrolyte temperature below 950.0°C during its production life is typically in the range of 0 to 30. A low temperature excursion count for a reduction cell is completed by summing the number of bath temperatures on the smelter knowledge base that are less than 950.0°C.

Output. Cell failure prediction - The single network output for this application specifies whether the reduction cell is in failure mode or not in failure mode, denoted by a 1.0 or 0.0, respectively. The neural network is used in this instance to identify an input data pattern as a criteria for cell failure or not cell failure, by specifying a 1.0 or 0.0 in each instance, respectively, during network training. If the trained network identifies an input data pattern as a failure condition it produces a 1.0 as an output, suggesting the reduction cell should be cut-out as it will tap-out as a consequence if left in production. On the other hand, a prediction of 0.0 indicates the cell should remain in production because it is not at risk of failure.

4.4 APPLICATION OF NEURAL NETWORKS TO PREDICT ELECTROLYTE TEMPERATURE

The application being scoped here involves the prediction of electrolyte temperature within an operating reduction cell at CABBL, using neural networks as an intelligent decision making methodology. It is intended that the prediction of electrolyte temperature will be possible by using related process parameters as inputs to a neural

network, trained using historical process data from the smelter knowledge base. It is important to note that the desired result from using a neural network for this particular application is to achieve a reduction in the associated production costs per unit of aluminium produced. However, prior to discussing the particularities of the electrolyte temperature prediction strategy, it is useful to introduce the reader to the characteristics of bath temperature and the significance of determining the value of this important process parameter.

4.4.1 Importance and Characteristics of Electrolyte Temperature

It has been stated previously that bath temperature is an important parameter in aluminium electrolysis. It is a significant indicator of how stable and efficiently a reduction cell is operating. Routine bath temperature measurements are taken to determine whether actions are required to return the reduction cell to an acceptable operating temperature range, shown previously as being 950.0 to 980.0°C at CABBL. It has been discussed that in smelting operations the addition of electrolyte additives, such as AlF_3 and Na_2CO_3, to the reduction cell is the major bath temperature control strategy. Hence, frequent additions of these important electrolyte additives to the reduction cell are necessary to maintain bath temperature within a predetermined acceptable control range.

In electrolytic processes in molten salts, such as the production of aluminium, the corrosive nature of the electrolyte, due to high sodium content and high temperature, produce associated problems with bath temperature measurement [194]. While temperature is one of the most important process control parameters, continuous measurements of bath temperature have not proven to be technically or economically viable. Remote sensors, such as optical pyrometers and measurements of the intensity of infra-red radiation, are not practical or accurate enough [11]. Both techniques involve the focusing of the measurement instrument on a representative section of the bath. However, the tendency of the bath to form a crust combined with the continuous agitation of the bath, due to the associated magnetohydrodynamics of the process, make it difficult to retain a representative uniform surface. In addition, measuring the intensity of infrared radiation is at its highest sensitivity at low temperatures, however, at the high temperatures used in aluminium reduction cells the sensitivity is significantly reduced [19]. Consequently, in order to measure the

bath temperature within a reduction cell, conventional thermocouples are commonly used for intermittent temperature measurements. While the most commonly used practical technique for measuring bath temperature in industrial aluminium electrolysis cells is by use of type K mineral insulated metal sheathed thermocouples, such thermocouples only last for a very short period, deteriorating rapidly each time they are placed in the bath, eventually being destroyed by the highly corrosive electrolyte [195]. In addition, this measurement technique has a further associated disadvantage in that the measurement is highly operator dependent. Due to the long response time, the signal will increase only very slowly near thermal equilibrium [196]. Hence, it is difficult to determine when the bath temperature is reached. However, while there are noted disadvantages associated with this bath temperature measurement technique, it is important to note that the measurement of bath temperature using type K thermocouples, immersed into the molten electrolyte at prescribed locations, is the most common measure of bath temperature used in the aluminium smelting industry. After each bath temperature measurement the thermocouple is removed from the electrolyte in order to limit the corrosive effects of the cell environment. While different aluminium industries have different procedures and frequencies for bath temperature measurements, rarely is the bath temperature measured more frequently than once per day or less often than weekly [196]. The main value of the intermittent temperature measurement is to provide information on average process conditions and cell stability and to give a warning of abnormalities within the reduction cell [13].

4.4.2 Existing Electrolyte Temperature Measurement Technique at CABBL

The existing technique used at CABBL for determining the bath temperature within an operating reduction cell involves manual measurement of the electrolyte temperature, using a type K thermocouple and associated data acquisition for recording bath temperature. However, as the thermocouple is required to be immersed in the molten electrolyte, it is often necessary beforehand for the process operator to use a suitable mechanism to manually remove a section of the hard crust within the reduction cell before the molten electrolyte is exposed. Following this procedure, the thermocouple is immersed into the molten electrolyte and the corresponding bath temperature is displayed on the data acquisition monitor, which the thermocouple is connected to, as shown in Figure 4.4.1.

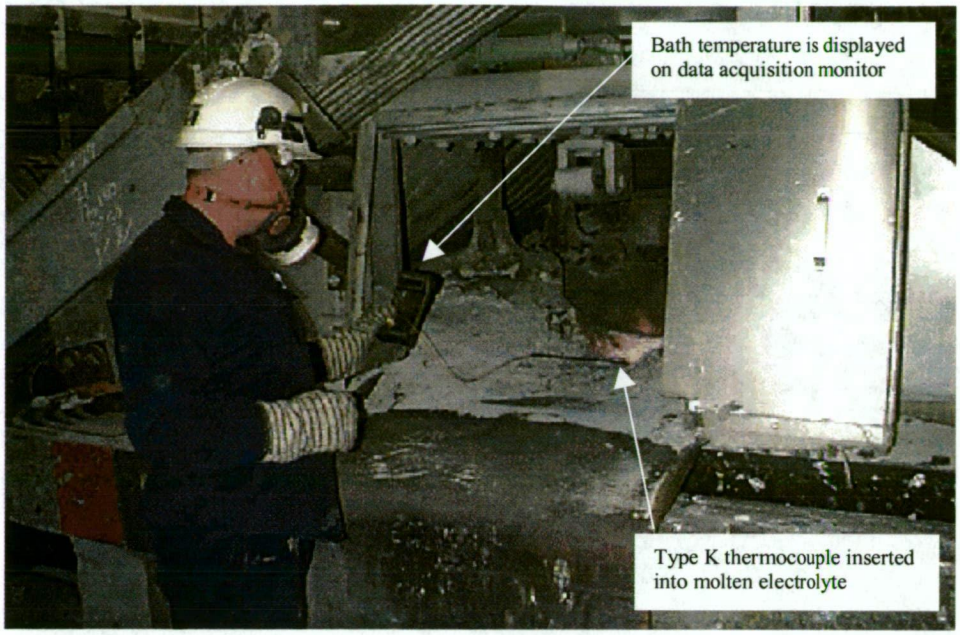


Fig. 4.4.1. Manual Bath Temperature Measurement Technique at CABBL Showing Thermocouple Immersion into Molten Electrolyte

After initial placement of the thermocouple in the molten electrolyte, the thermocouple remains in the bath for a period substantial enough to ensure the displayed temperature on the data acquisition monitor does not change significantly. That is, the temperature displayed on the data acquisition monitor increases initially after placement of the thermocouple in the bath until such a time that the thermocouple heats to the same temperature as the electrolyte, hence, the displayed temperature on the data acquisition monitor does not change once this condition is achieved. Consequently, this temperature is recorded by the process operator as the bath temperature for the corresponding reduction cell. This procedure is completed for each of the 540 reduction cells at CABBL every 48.0 hours. In addition to these manual temperature measurements, the process operator is also required to enter the recorded temperatures into the smelter knowledge base. Hence, it is evident that the procedure for manual bath temperature measurements requires significant labour content and occupies a large majority of routine cell maintenance. Subsequent to manual bath temperature measurements, corrective action is administered to each reduction cell as appropriate to return the bath temperature to the target of 965.0°C. It is interesting to note that an experimental investigation [197] has shown the error associated with manual temperature measurements at CABBL is approximately ±4.0°C. It has been shown that this error is evenly attributed to equipment error and

operator error. This error value is a direct indication of the error associated with the existing bath temperature measurement technique. It is useful to compare this error value with the error associated with the neural network modelling strategy that will be completed for predicting bath temperature.

4.4.3 Rationale for Prediction of Electrolyte Temperature Using Neural Network Modelling

Manual bath temperature measurements at CABBL represent a significant portion of routine cell operations. In addition to manual temperature measurements, associated activities such as bath temperature recording for each reduction cell and data entry into the smelter knowledge base require significant time to complete. Bath temperature measurements are completed every 48.0 hours per reduction cell, or bi-daily, requiring approximately 1.0 minute per reduction cell for a total of 540 reduction cells in operation at the plant. In addition, recording the measured temperatures and entering the recorded values into the plant knowledge base requires approximately an additional 5.0 seconds per reduction cell. Hence, the labour content required for manual bath temperature measurements and recording is given by the following:

$$\begin{aligned} \text{labour} &= [\text{measurement time} + \text{recording time}] \\ &= [1{,}642.5 + 136.9] \text{ hours per year} \\ &= 1{,}779.4 \text{ hours per year} \end{aligned} \tag{4.4.1}$$

Thus, approximately 1,779 hours per year are required to complete manual bath temperature measurements and recording, representing a substantial labour requirement. Hence, the reduction or elimination of manual bath temperature measurements would result in significant labour savings.

While the associated labour savings are highlighted it is also important to note that there are substantial health and safety issues associated with manual bath temperature measurements. The environment in which bath temperature measurements are completed is extremely hot and unpredictable. While precautions are taken to reduce the risk of exposure to molten electrolyte, through the provision of appropriate safety apparatus, molten electrolyte splashing onto process operators is directly responsible

for occasional burn injuries, resulting in health and safety issues. In addition, there is a significant cost associated with thermocouple consumption during bath temperature measurements. The corrosive environment within the reduction cell yields deterioration and ultimate failure of the thermocouples. Consequently, thermocouple consumption costs are substantially high and contribute to the cost of aluminium production. Therefore, the economic benefit of implementing neural networks to accurately predict bath temperature is reduced labour requirement, reduced associated health and safety issues and reduced equipment consumption.

4.4.4 Specification of Neural Network Inputs for Electrolyte Temperature Prediction Application

The parameters used as network inputs for this particular application are measured parameters of the Hall-Heroult process. The parameters specified here include any variables that are considered to have at least some correlation with bath temperature. While some of the process parameters selected in the first instance may not contribute to the prediction of bath temperature, neural network modelling will identify non-contributing inputs that can be omitted. However, similar to the previous applications discussed, it is important in the first instance to include all parameters that may have an influence on the prediction of bath temperature and eliminate those parameters from the model that are identified as non-contributing inputs. The process parameters selected as network inputs in this instance are shown in an illustration of the neural network model used for this particular application, Figure 4.4.2.

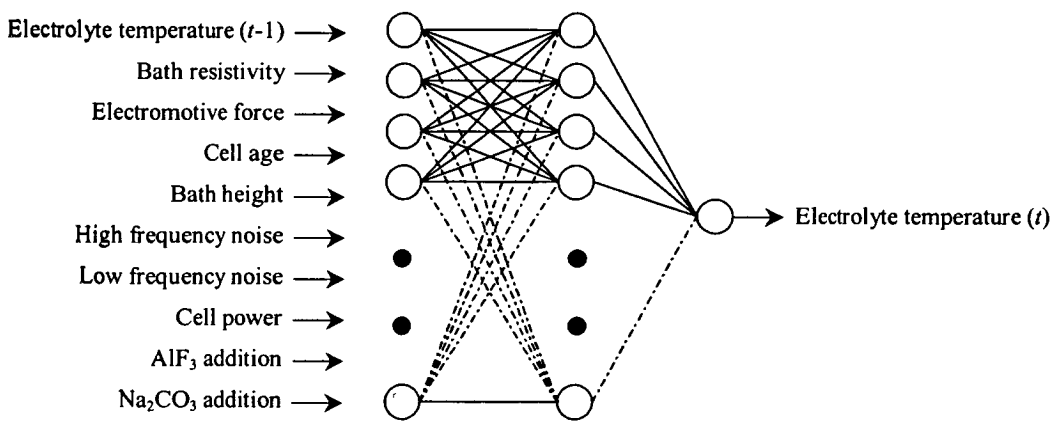


Fig. 4.4.2. Illustration of Neural Network Model for Electrolyte Temperature Prediction Application

The following section gives a brief description of each process parameter used in this instance and provides a justification for its selection as a network input for electrolyte temperature prediction. While the process parameters have been previously discussed for the preceding applications, it is necessary to reiterate each input variable here in order to highlight the justification for its inclusion as a neural network input variable for electrolyte temperature prediction. Further, the majority of the process variables are highlighted in Figure 4.4.3 to illustrate their role in the reduction cell.

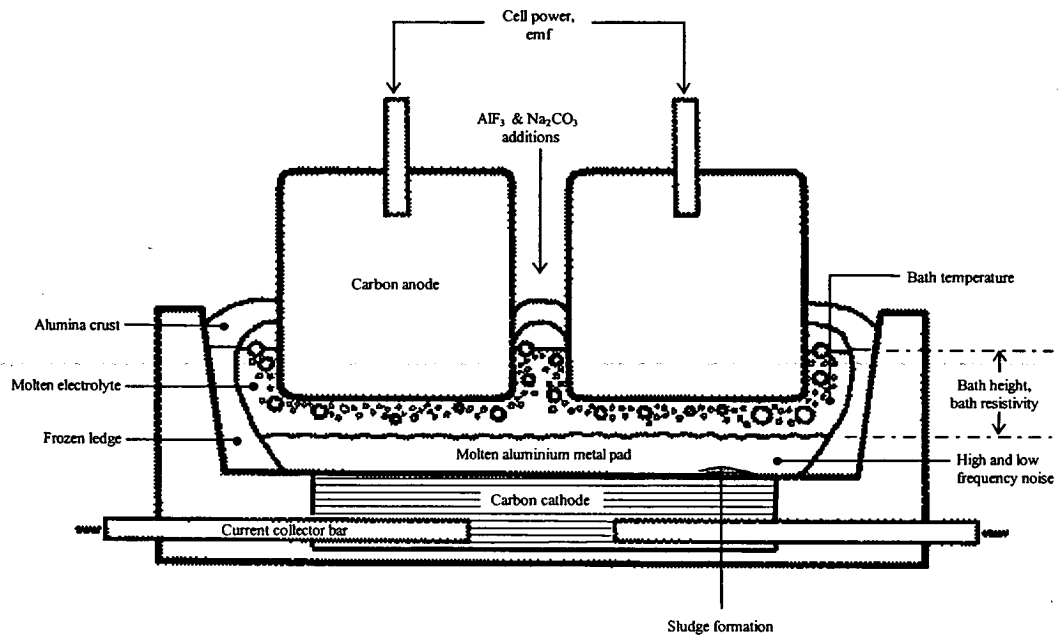


Fig. 4.4.3. Cross-Section of Reduction Cell Indicating Process Parameters used as Neural Network Inputs for Electrolyte Temperature Prediction Application

1. Electrolyte temperature (t-1) - Electrolyte temperature for use as a network input in this instance is the previous bath temperature recorded for each reduction cell. While the output of the network is the current bath temperature for a designated time, the previous bath temperature is the last recorded bath temperature for the reduction cell in which the bath temperature prediction is being completed. It is useful as a network input as it indicates the most recent available bath temperature measurement. While initially the previous bath temperature will be the manual completed measurement, the previous bath temperature will ultimately be a predicted value once the neural network is implemented. In particular, the predicted bath temperature at time t-1 will become the previous bath temperature for the prediction at time t. While the manual bath temperature measurement technique has been previously discussed it is also

noted that frequent bath temperature measurements are recorded to the smelter knowledge base. Typical values of electrolyte temperature are in the range of 935.0 to 995.0°C.

2. Bath resistivity - From published data [183,184] regarding the electrical conductivity of the Hall-Heroult electrolyte it is shown that the electrical conductivity of the bath increases with increasing AlF_3 content, decreasing bath height and decreasing bath temperature. Hence, bath resistivity is low for high AlF_3 content and therefore is an important indicator of AlF_3 content present in the bath chemistry, which has been shown to be closely related to bath temperature. In addition, bath resistivity will be low when the cell has low bath height and low bath temperature [185]. It has been previously noted that bath resistivity measurements are recorded automatically to the smelter knowledge base. Typical values of bath resistivity are in the range of 4.2 to 5.2ohms/mm.

3. Electromotive force - A graphical analysis has shown that emf is generally high for low bath temperature. Hence, emf is included as a network input in this selection stage as it may be a useful contributor to bath temperature prediction. It has been previously noted that emf for each reduction cell is measured using combined automated and manual techniques and subsequently documented on the smelter knowledge base.

4. Cell age - It has been noted that bath temperature is slightly higher on average in reduction cells with high age. Hence, cell age is included as a network input variable in this instance as it is considered to have some relationship with electrolyte temperature. It has been noted that cell age is in the range 0 to 2,500 days.

5. Bath height - Bath height is an important parameter as it indicates the volume of bath in the reduction cell. A study of changing bath height with changing bath temperature has shown bath temperature to be proportional to bath height. Specifically, high bath temperature is associated with high bath height while low bath height typically yields low bath temperature. While the manual bath height measurement technique has been previously discussed it is also noted that frequent

bath height measurements are recorded to the smelter knowledge base. Bath height is typically in the range of 180.0 to 250.0mm.

6. High frequency noise - It has been noted that high frequency noise is an indication of instability within the reduction cell. High frequency noise is usually attributed to anode problems, such as high anode consumption rate, anode spike, which is a carbon lump protruding from the anode at a site where a carbon fragment or lump of sludge has previously been located, and low anode placement within the reduction cell. Consequently, each of these listed anode conditions has a correlation with bath temperature. In particular, high carbon consumption rate, anode spikes and low anode placement yield high bath temperature. Hence, high frequency noise is a useful network input for the estimation of electrolyte temperature. It has been noted that high frequency noise values for each reduction cell are frequently recorded to the smelter knowledge base using an automated technique and are typically in the range of 0.02 to 0.05μohms/s.

7. Low frequency noise - Low frequency noise is generally attributed to the movement of the entire metal pad as a result of sludge on the cathode. Moreover, sludge is generally formed as a result of low bath temperatures, hence, low noise may be useful as an indication of the value of bath temperature in this instance. Hence, low frequency noise is a useful network input. Low frequency noise values are recorded to the smelter knowledge base using an automated technique and are typically in the range of 0.05 to 0.25μohms/s.

8. Cell power - Power is an important network input as it specifies the amount of energy being supplied to the cell for aluminium production. A study of changing cell power with bath temperature variation has shown that a low bath temperature requires the application of additional voltage to increase bath temperature to the acceptable control limits, consequently cell power increases. Conversely, high bath temperatures require the reduction of power to the reduction cell, consequently cell power decreases. Hence, power is a useful indicator of the temperature of the reduction cell and is therefore included as a network input in this instance. Cell power for each reduction cell is recorded to the smelter knowledge base

automatically using appropriate data acquisition and associated equipment. Typical cell power values are in the range of 420.0 to 470.0kW.

9. AlF_3 addition - AlF_3 addition is a necessary input to the neural network as it specifies the mass of AlF_3 added to the cell, which has a significant influence on bath chemistry and therefore, bath temperature. For cells that have consistently high bath temperature, AlF_3 additions will be high and likewise, low bath temperatures require low additions of AlF_3. Hence, AlF_3 addition is a necessary input to the network as it has a noticeable influence on the bath temperature within the reduction cell. All AlF_3 additions to each reduction cell are recorded on the smelter knowledge base. Typical additions of AlF_3 are in the range of 0.0 to 70.0kg.

10. Na_2CO_3 addition - Similarly, Na_2CO_3 addition is a necessary input to the network as it specifies the mass of Na_2CO_3 added to the cell in previous bath additions, which also influences bath chemistry and therefore, bath temperature. Specifically, high additions of Na_2CO_3 indicate low bath temperature, while high bath temperatures require no Na_2CO_3 to be added. Hence, Na_2CO_3 addition is a necessary input to the network as it has an influence on bath temperature. All Na_2CO_3 additions to each reduction cell are recorded on the smelter knowledge base. Typical additions of Na_2CO_3 are in the range of 0.0 to 70.0kg.

Output. Electrolyte temperature (t) - Electrolyte temperature, shown as being the temperature of the molten electrolyte, is the single output of the neural network for this application and is the estimated temperature of the electrolyte at time t. The typical range of values for this parameter is 935.0 to 995.0°C.

4.5 CONCLUDING REMARKS

The objective of applying neural networks to the applications discussed is to achieve a reduction in the associated costs of aluminium production at CABBL. For each application, the selected neural network models will be applied using the specified network input and output variables in each instance. It has been noted, however, that the specified input variables are potential neural network inputs only. Moreover, the required input variables for each application will be determined during neural network modelling, using the predictive and casual importance techniques described

previously for determining the contribution of process parameters towards the network predictions. The network modelling completed in the sensitivity analysis, documented in an earlier chapter, was useful and necessary to study the behaviour of the developed neural network models and introduce and investigate the predictive and casual importance techniques. However, neural network modelling for the listed applications in this instance is particularly beneficial as it will highlight the possibility of using neural networks at the Bell Bay smelter to achieve economic benefit. In addition, it will also be useful to further understand and determine relationships that exist among parameters of the Hall-Heroult process. Hence, the following chapter documents the results of neural network modelling for each of the three listed applications, including the accuracy and ordering of input parameter importance obtained by the different network models and the computation time required by each paradigm.

CHAPTER FIVE

Assessment of Neural Network Models for Industrial Applications

5.1 EXPERIMENTAL PROCEDURE FOR NEURAL NETWORK MODELLING

Prior to discussing the neural network modelling results, it is important to note that the modelling was completed for each of the studied industrial applications by following the procedure outlined in the following:

i). data acquisition and formatting
ii). data pre-processing
iii). model specification
iv). model training and testing
v). model analysis

i). Data acquisition and formatting - Historical process data was retrieved from the smelter knowledge base to produce a data file for the process variables to be modelled, for each of the studied applications. While the majority of the data was used for network training purposes, approximately 20.0% of the data was used for network testing in each instance. The data was carefully selected for each application to ensure the entire range of operating conditions for each network model variable was included in the data set. Further, an even distribution of data was targeted for each application to improve network prediction accuracy over the entire data range.

ii). Data pre-processing - In order to prepare the process data for network modelling, specific mathematical transforms were required. In particular, the formula documented in Equation 2.2.7 was applied to scale the parameters in the data patterns between the bounded range 0.0 and1.0.

iii). Model specification - The particular neural network models applied and investigated in the sensitivity analysis were applied to each of the industrial applications being studied here. It is useful to restate that the particular neural networks applied are:

- Widrow-Hoff (WH) Neural Network
- Backpropagation - 1 hidden layer - (BP1) Neural Network
- Backpropagation - 2 hidden layers - (BP2) Neural Network
- Radial Basis Function (RBF) Neural Network
- Radial Basis Function - incorporating Kohonen - (RBFKOH) Neural Network
- General Regression Neural Network (GRNN)

While there is substantial literature available in regard to the characteristics and particularities of the majority of the neural networks listed, it is useful to re-iterate that the RBFKOH model is a combination of the radial basis function and Kohonen neural networks. Specifically, the Kohonen neural network is used to cluster similar data patterns prior to training the radial basis function model, in an attempt to improve the performance of the radial basis function neural network. For each of these listed neural network models a particular range of network architectures are used in each instance in order to establish the network architecture that yields minimum prediction error for each applied model for each studied application. In addition, where appropriate it is also necessary to vary the value of the adjustable parameters used in the activation function of the different networks in order to minimise prediction error. In particular, the receptive field width, σ, used in the activation functions of the RBF, RBFKOH and GRNN models was varied over a suitable range, shown previously as 0.1 to 0.9. Further, it is important to note that the learning rate, α, used in the delta rule for the weight update procedure was decreased during network training from a large initial value of 0.9 to a minimum value of 0.1 as iterations increased. The purpose of this was to optimise the convergence speed of the network, allowing large weight changes when the prediction error is large, decreasing the size of the weight change as the prediction error approaches a minimum, to allow convergence. It is useful to note here that this technique was adopted for this purpose for all of the applied neural networks that use the delta rule

for the weight update procedure. In particular, these networks are the WH, BP1, BP2, RBF and RBFKOH models.

iv). Model training and testing - For each application, the data file prepared as a result of data acquisition, formatting and pre-processing was presented to each of the neural networks. Following completion of a sufficient number of iterations, where appropriate, for each neural network model, a set of train and test RMS error values are obtained for each application. These RMS error values are then used in the model analysis stage of this procedure.

v). Model analysis - A particular procedure is adhered to in this instance for each of the studied applications in order to complete an analysis of the developed neural network models. It is useful to document this procedure here.

Step 1. For each neural network model, determine the network architecture that produces minimum prediction error, based on a study of RMS error.

Step 2. Using the network architecture that produces minimum RMS error, for each network model, use the predictive importance technique to determine the contribution of each input variable on the network prediction. Subsequently, remove any non-contributing variables from each of the studied network models and re-train the models using the remaining input variables to obtain a RMS error value for the train and test data sets.

Step 3. Complete a sensitivity analysis, using the casual importance technique, to quantitatively determine the degree of importance of each contributing input variable in the network model.

Using this methodical procedure, neural network modelling is completed for each of the studied industrial applications. The major findings from this network modelling are documented in the following section. It is useful to note here that the neural network modelling results are presented consecutively, in defined sections, for each of the studied applications. In addition, for comparison with the neural network modelling results, a multi-variable regression analysis, MVRA, was completed on the training data for each of the industrial applications studied. Further, it is useful to note that the MVRA is completed using the specified input parameters as source

variables and the specified output parameters as response variables, in each instance. While the results of the MVRA allowed a RMS error value to be determined for the train and test data sets in each instance, the percentage contribution of the input variables is also completed for each application, which is shown to be based on the magnitude of the regression coefficients obtained.

5.2 ASSESSMENT OF NEURAL NETWORK MODELS FOR ELECTROLYTE ADDITIVE PREDICTION

It has been noted previously that there are 12 process parameters selected as potential neural network inputs for this electrolyte additive prediction application. Further, the quantity of two particular electrolyte additives to add to the reduction cell form the outputs of this model. It is useful here to highlight the particular process parameters used in the neural network models, as shown in Figure 5.2.1, using an arbitrary neural network architecture.

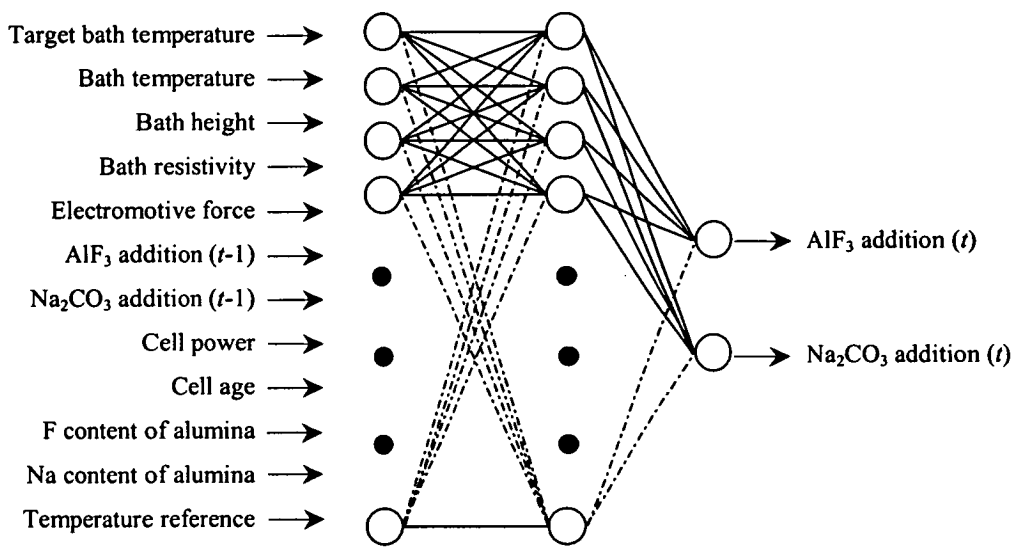


Fig. 5.2.1. Illustration of Neural Network Model for Electrolyte Additive Prediction Application

Data acquisition from the smelter knowledge base with subsequent data formatting and pre-processing produced 1,565 data patterns in total for this industrial application. Data preparation formed a complex stage of network modelling, including the elimination of corrupt information from the acquired data, ensuring sufficient data for each model variable was included in the data patterns. In addition, it was necessary to cover the entire operating range for each parameter and scale each

variable between the minimum and maximum limits of its operating range. The processed data was then divided into 1,365 training data patterns, with the remaining 200 patterns used as test data. The train and test data sets were carefully selected to ensure the network output variables were well represented in each data set, covering the entire range of values. It is shown in Figures 5.2.2(a) and (b) that an approximately uniform distribution of data for the network output variable AlF_3 is obtained for the train and test data sets. However, it is shown for higher values of AlF_3, 63.0kg or greater, that the output variable is not as well represented in the train and test data patterns. This is attributed to the fact that additions of this magnitude for this electrolyte additive are extremely rare, hence, limiting the available information on the smelter knowledge base. Nevertheless, the most significant portion of the operating range is well represented, with some examples included for higher additions of AlF_3. On the other hand, the distribution of Na_2CO_3 is not so well represented in the train and test data sets, as shown in Figures 5.2.2(c) and (d).

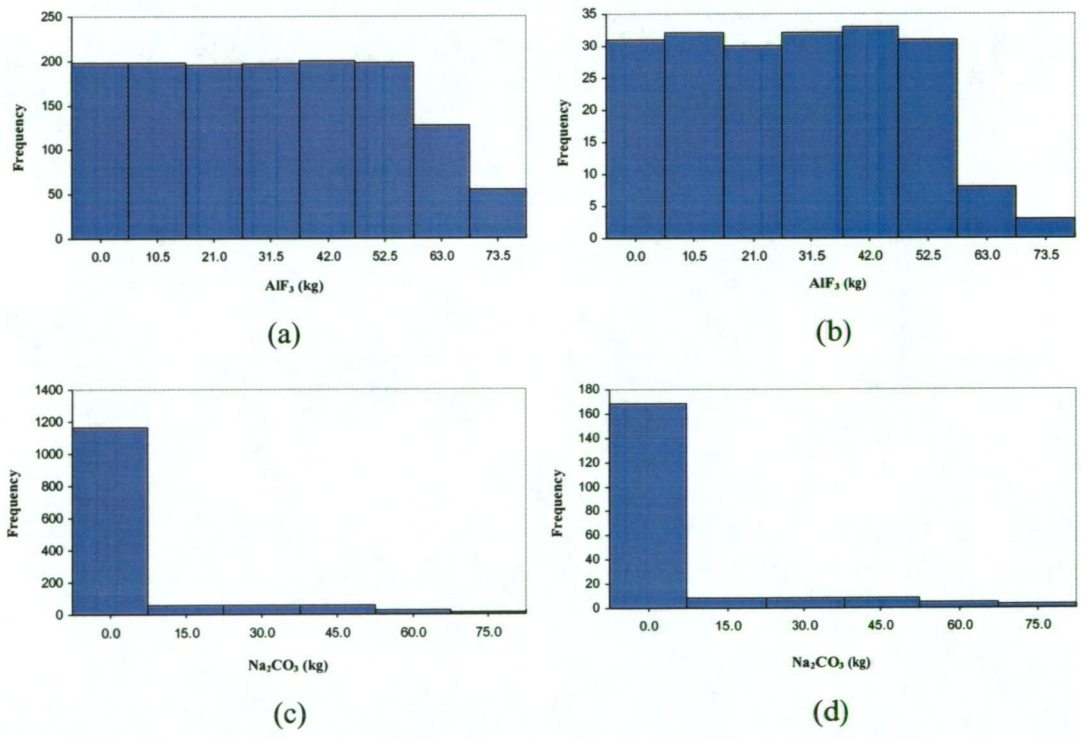


Fig. 5.2.2. Histogram Showing Distribution of the Network Output Variables in the Data Sets. (a) Training - AlF_3, (b) Test - AlF_3, (c) Training - Na_2CO_3, and (d) Test - Na_2CO_3

It is important to note that AlF_3 and Na_2CO_3 are not simultaneously added to a reduction cell at any one instant, rather, either only AlF_3 or Na_2CO_3 is added. Consequently, a data pattern containing an AlF_3 addition cannot also contain a Na_2CO_3 addition, and vice versa. Hence, the number of data patterns containing information for 0.0kg Na_2CO_3 addition is equivalent to the combined number of data patterns containing information for an AlF_3 addition of 10.5kg or greater. Conversely, the number of data patterns containing information for 15.0kg Na_2CO_3 addition or greater is equivalent to the number of data patterns containing information for an AlF_3 addition of 0.0kg. Hence, the distribution of the data patterns for the output variable Na_2CO_3 is governed by the distribution of the AlF_3 data patterns. It is important to note that AlF_3 has precedence in the data patterns because it is the most significant electrolyte additive.

It is useful to consider the results of the MVRA completed on the developed training and test data sets, as shown in Table 5.2.1, prior to considering the neural network modelling results. It is noted in Chapter Three that the MVRA is a multi-variable regression analysis that incorporates specific mathematical techniques to develop a linear regression equation that is used to predict the value of one or more response variables based on the specification of a source variable vector. In order to determine the significance of the source variables in the developed MVRA model, a statistical *t*-test is applied. A *t*-critical value of 1.645 is applicable in this instance, based on a sample population of 1,365, 12 source variables and a confidence level of 95.0%. Hence, the magnitude of the *t*-statistic is required to be higher in magnitude than 1.645 for a regression coefficient to be significantly different to zero. Comparing the *t*-statistic value for each parameter with *t*-critical identifies the significant source variables in the MVRA model, as shown in Table 5.2.1.

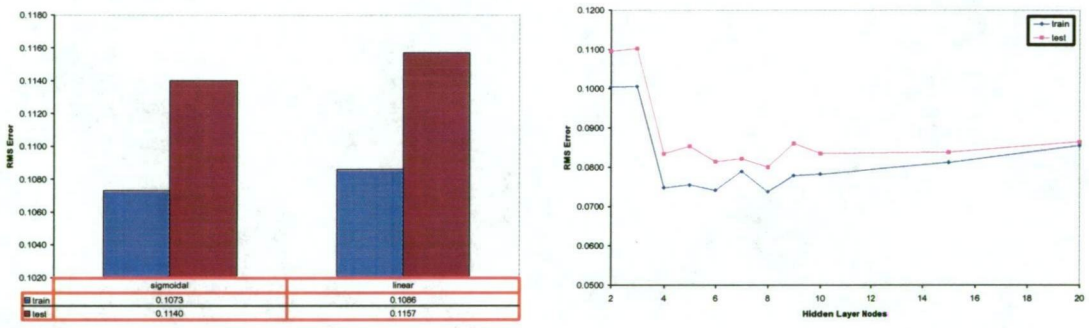
TABLE 5.2.1. Multi-Variable Regression Analysis Results for Electrolyte Additive Prediction Application

Regression Statistics			
r: 0.7821	r^2: 0.6117	*adjusted* r^2: 0.6082	*observations*: 1,365
Parameter	Regression Coefficient	*t*-statistic	Significant
target temperature	0.37837	8.42750	yes
bath temperature	1.03565	16.52537	yes
bath height	0.00127	0.32563	no
bath resistivity	-0.73797	-13.67482	yes

emf addition	-0.14341	-3.42944	yes
AlF_3 add. (t-1)	0.11230	4.74401	yes
Na_2CO_3 add. (t-1)	-0.00941	-0.54080	no
cell power	0.25630	3.70546	yes
cell age	-0.00445	-0.16730	no
F content	-0.00526	-0.67932	no
Na content	0.00042	0.00443	no
temperature ref.	0.00998	1.25384	no

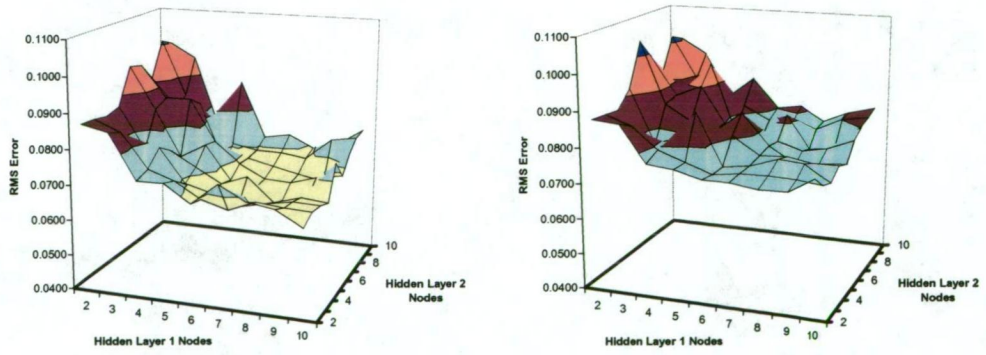
The regression statistics for this application have shown an *adjusted* r^2 value of 0.6082 is achieved, highlighting that approximately 60.8% of the variation of the output variables is explained by the variation of the source variables. Consequently, approximately 39.2% of the variation in the output variables is unexplained by this particular regression model. In addition, it can be seen from the results of the MVRA that target temperature, bath temperature, bath resistivity, emf, AlF_3 addition (t-1) and cell power are the only contributing input variables in the regression model. It is shown that the remaining variables are not contributing to the prediction of electrolyte additive quantity. This is highlighted by the comparatively low magnitude of the associated regression coefficients for the non-contributing variables and further by the t-statistic value, which is shown to be lower than the t-critical value in each instance. The percentage contribution of the input variables can be determined from the MVRA by considering the value of the regression coefficient for each contributing parameter. Similar to the technique described for the sensitivity analysis completed in a previous chapter, the percentage contribution of the input variables is calculated by summing the absolute values of the regression coefficients and dividing the value of each regression coefficient for each parameter by the sum. This yields a percentage contribution value for each contributing input variable. It has been shown that the most significant parameter in the MVRA model is bath temperature (38.9%), followed by bath resistivity (27.7%), target bath temperature (14.2%), cell power (9.6%), emf (5.4%) and AlF_3 addition (t-1) (4.2%). In addition, comparing the actual and predicted values of AlF_3 and Na_2CO_3 for the train and test data sets assesses the accuracy of the MVRA model for this application. This is completed in this instance to obtain RMS error values of 0.1081 and 0.1129 for the train and test data sets, respectively. It will be useful to compare the MVRA results with the findings from the neural network modelling, the results of which are detailed in the following.

For the six neural network models applied to this application, RMS error behaviour for the train and test data sets with changing network architecture is shown in Figures 5.2.3 to 5.2.7, inclusive, and tabulated in Appendix C, Tables C.1 to C.6, inclusive. It is shown by the general trend exhibited by each of the applied models that RMS error changes significantly with changing architecture. Nevertheless, it is shown for a particular architecture in each instance that a relatively low RMS error value is achieved by each of the applied neural networks. An assessment of the practical implications of the RMS error achieved is provided later in this section. However, it is useful to note here that the general trend exhibited by each model and the minimum RMS error achieved in each instance confirm the ability of the applied neural networks to accurately model the Hall-Heroult process for electrolyte additive predictions.



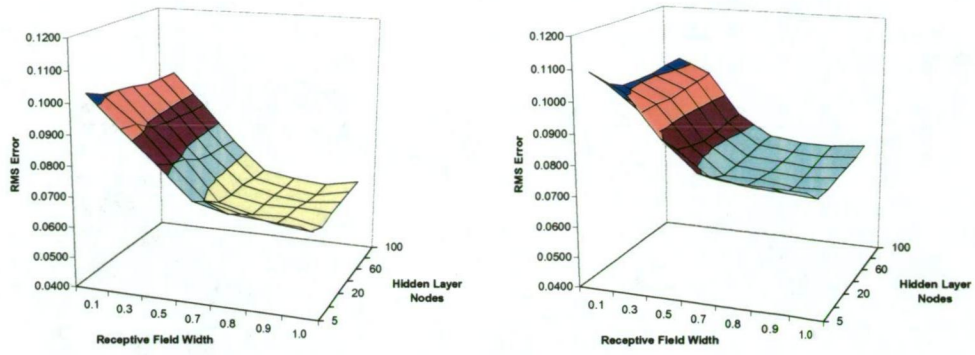
(a) (b)

Fig. 5.2.3. RMS Error Behaviour with Changing Architecture. (a) WH Network, and (b) BP1 Network



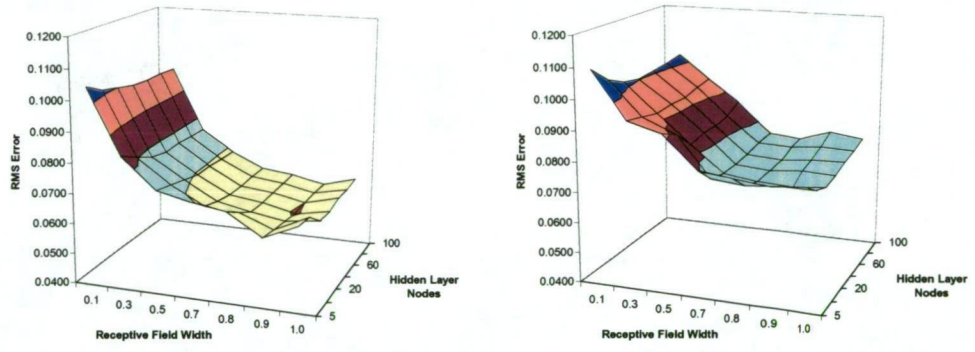
(a) (b)

Fig. 5.2.4. BP2 Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



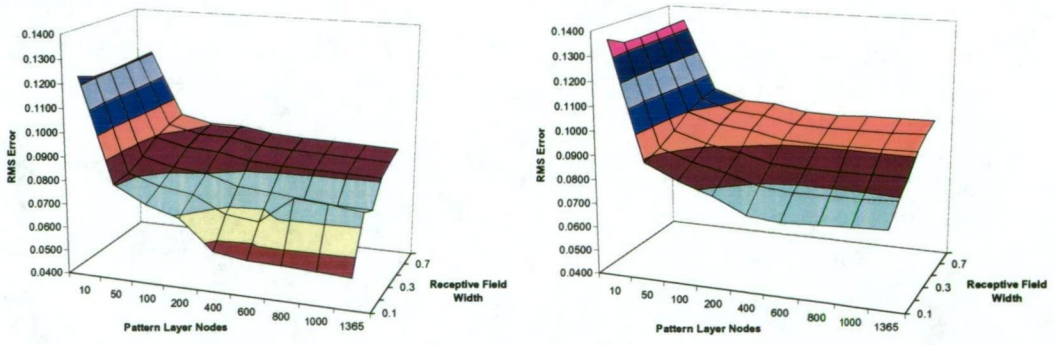
(a) (b)

Fig. 5.2.5. RBF Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.2.6. RBFKOH Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.2.7. GRNN RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

Considering firstly the WH neural network, the network prediction error was compared in this instance for the sigmoidal and linear summation activation function

in the output layer of the network. It is shown that the RMS error values associated with the sigmoidal activation function are 0.1073 and 0.1140 for the train and test data sets, respectively, which are lower than the RMS error values of 0.1086 and 0.1157 for the train and test data sets, respectively, achieved using the linear summation activation function. While 1,000 iterations were completed for the WH network in each instance, it was found from a study of RMS error behaviour with increasing iterations that while the RMS error decreased rapidly in the initial 30 iterations, after approximately 200 iterations the RMS error did not change for the train and test data sets, highlighting convergence of the network weights. However, an improved RMS error from that of the WH network has been achieved using the BP1 network for this particular application. It is shown that 8 hidden layer nodes yield minimum RMS error for the BP1 model, shown to be 0.0737 and 0.0800 for the train and test data sets, respectively. This is achieved using the sigmoidal activation function in the hidden and output layer nodes. It is important to note that while the results from using the linear summation activation function are not presented here, it was found that the sigmoidal function produced minimum RMS error. It is useful to note that while 1,000 iterations were completed for this network model, it was found that approximately 650 iterations were required for convergence of the network weights, after which RMS error was shown not to change for the train and test data sets. However, RMS error is further improved using the BP2 network. It is shown that a RMS error of 0.0615 and 0.0719 is achieved for the train and test data sets, respectively, using the BP2 model. It is shown that this is the minimum RMS error that can be achieved using the BP2 network, obtained using a network architecture of 8 and 6 nodes in the first and second hidden layers, respectively, while the sigmoidal activation function was used in both hidden layers and also in the output layer. It can be seen that RMS error varies significantly with changing number of hidden layer nodes, highlighting the need to study RMS error behaviour with increasing number of hidden layer nodes in order to maximise network accuracy. A study of RMS error behaviour over a period of 1,000 iterations has shown that approximately 800 iterations are necessary and sufficient for convergence of the network weights in the BP2 model. Applying the RBF network, which has been discussed as having the input to hidden layer weights selected randomly from the training data patterns, has shown a lower training and test RMS error than that obtained using the WH and BP1 networks, while slightly higher than the BP2 network. It is shown that minimum

RMS error, shown to be 0.0635 and 0.0730 for the train and test data sets, respectively, is obtained using 20 hidden layer nodes and a σ value of 0.9. This result is achieved using the Gaussian function as the activation function in the hidden layer nodes, while a linear summation function is used as the activation function in the output layer nodes. Studying RMS error behaviour over a period of 1,000 iterations has shown that minimum RMS error is achieved at approximately 250 iterations, after which RMS error does not change, highlighting convergence of the network weights. However, using the RBFKOH model has shown an improvement in RMS error for the train and test data sets, compared to the RBF network. In particular, it is shown that 20 hidden layer nodes and σ equal to 0.9 yields a RMS error of 0.0607 and 0.0716 for the train and test data sets, respectively. It is interesting to note that this particular test RMS error is lower than that achieved with the WH, BP1, BP2 and RBF networks, highlighting that the RBFKOH model has the best generalisation ability for this particular application compared to the other models considered so far. Similar to the RBF network, the Gaussian function is used as the activation function in the hidden layer nodes, while the linear summation function is used as the activation function in the output layer nodes. While 1,000 iterations were used to train the RBFKOH network initially, a study of RMS error behaviour with increasing iterations has shown the RBFKOH network weights to converge after approximately 250 iterations. However, the GRNN model has shown the lowest RMS error for the train and test data sets for this particular application. It is shown that a RMS error of 0.0549 and 0.0705 for the train and test data sets, respectively, is achieved using 600 pattern layer nodes and σ equal to 0.1. This is achieved using the exponential function in the pattern layer nodes while a linear summation function is used in the summation layer nodes, while the output layer nodes complete a division of the numerator and denominator nodes. It is useful here to restate that the GRNN trains in a single pass of the training data, hence, only a single iteration is required for network training.

An ANOVA study is completed to evaluate the statistical significance of the difference in error associated with each model. It has been noted that the most appropriate statistical method for comparison of multiple population means to determine the statistical significance of differences is ANOVA. However, it is noted that while ANOVA highlights a statistically significant difference between two or

more population means, it does not indicate which particular population means are different. However, subsequent to ANOVA, a series of statistical *t*-tests can be applied to determine which population means are statistically significantly different, as detailed in Chapter Two. It is important to note that the normalised values of actual and predicted AlF_3 and Na_2CO_3 additions are used for this statistical analysis. It is not necessary to convert the normalised values to their original magnitude, as the variance of the data, which is used to calculate the test statistic, is a proportional measure. Hence, the test statistic calculated is the same for the original and normalised data. Further, as there are two output variables associated with this particular application then it follows that there is an error associated with the prediction of AlF_3 and error associated with the prediction of Na_2CO_3. Hence, it is necessary to complete the ANOVA study for the error associated with each output parameter. In addition, there are 7 models applied to this application, hence, there are 7 populations of size 200, as the test data set error is used for this statistical evaluation. A critical test statistic value of 2.1051 is appropriate for ν_1 equal to 6 and ν_2 equal to 1,393, using a significance level, α, of 0.05. The results of the ANOVA study for this application are documented in Table 5.2.2. The null hypothesis for this investigation is that the population means are equal, while the alternative hypothesis is that at least two population means are unequal. The null hypothesis is accepted if F is less than F_α.

TABLE 5.2.2. Test Data Set Error ANOVA Results for Electrolyte Additive Prediction Application

ANOVA Statistics			
Aluminium Fluoride - AlF_3			
MST: 0.0715	*MSE*: 0.0058	*F*: 12.3847	F_α: 2.1051
Soda-Ash - Na_2CO_3			
MST: 0.0785	*MSE*: 0.0061	*F*: 12.8689	F_α: 2.1051

	AlF_3		Na_2CO_3	
Population	Mean	Variance	Mean	Variance
p_1 - (WH)	0.0014	0.0095	0.0018	0.0089
p_2 - (BP1)	0.0010	0.0094	0.0012	0.0108
p_3 - (BP2)	0.0013	0.0150	0.0017	0.0156
p_4 - (RBF)	0.0018	0.0118	0.0014	0.0137
p_5 - (RBFKOH)	0.0021	0.0095	0.0019	0.0097
p_6 - (GRNN)	0.0008	0.0087	0.0011	0.0091
p_7 - (MVRA)	0.0029	0.0128	0.0027	0.0147

The ANOVA study has shown that F is greater than F_α in each instance. It is shown that the test statistic has a value of 12.3847 and 12.8689 for AlF_3 and Na_2CO_3, respectively, compared to the critical value of 2.1051 in each instance. Hence, the null hypothesis is rejected and it is accepted that at least two population means are statistically significantly different in each instance. Therefore, completing a statistical t-test for each paired population will highlight those population means that are statistically significantly different. The results of the statistical t-tests are shown in Table 5.2.3, noting that there are $(p)(p\text{-}1)/2$ t-tests to complete for each output parameter, where p is equal to 7 in this instance, corresponding to the 7 models applied. Similar to the sensitivity analysis, the significance level is selected as 99.0%, or α equal to 0.01, to minimise the probability of a Type I error occurring. Hence, the t-critical value for this analysis is determined to be 2.345 for a degree of freedom value of 199. It is useful to re-iterate here that the two populations considered in each statistical t-test are identified by indices. For example, a statistical t-test comparing population 1, p_1, and population 2, p_2, is denoted a t_{12}.

TABLE 5.2.3. Test Data Set Error Statistical t-Test Results for Electrolyte Additive Prediction Application

t_1 to t_7	t_2 to t_7	t_3 to t_7	t_4 to t_7	t_5 to t_7	t_6 to t_7
Aluminium Fluoride - AlF_3					
$t_{12} = 7.852$	$t_{23} = 8.651$	$t_{34} = 5.321$	$t_{45} = 2.947$	$t_{56} = 5.327$	$t_{67} = 8.954$
$t_{13} = 6.298$	$t_{24} = 9.527$	$t_{35} = 12.302$	$t_{46} = 5.618$	$t_{57} = 6.058$	
$t_{14} = 5.210$	$t_{25} = 11.234$	$t_{36} = 21.435$	$t_{47} = 9.624$		
$t_{15} = 19.214$	$t_{26} = 8.065$	$t_{37} = 11.818$			
$t_{16} = 11.230$	$t_{27} = 12.068$				
$t_{17} = 9.504$					
Soda-Ash - Na_2CO_3					
$t_{12} = 8.351$	$t_{23} = 5.057$	$t_{34} = 13.458$	$t_{45} = 2.947$	$t_{56} = 11.274$	$t_{67} = 6.216$
$t_{13} = 11.284$	$t_{24} = 21.571$	$t_{35} = 18.910$	$t_{46} = 6.350$	$t_{57} = 2.807$	
$t_{14} = 12.507$	$t_{25} = 19.913$	$t_{36} = 9.066$	$t_{47} = 8.275$		
$t_{15} = 6.328$	$t_{26} = 7.624$	$t_{37} = 17.279$			
$t_{16} = 2.984$	$t_{27} = 5.819$				
$t_{17} = 8.333$					

The results of the statistical analysis have shown that the null hypothesis is rejected for all paired populations, as the t-statistic value is greater in magnitude than the t-critical value in each instance. Hence, the alternative hypothesis, that the means of the paired populations are statistically significantly different, is accepted for all

paired populations. This confirms that the error associated with each of the applied models is statistically significantly different, which is an important finding as it means that no two models produce statistically equivalent error.

While RMS error has been documented for the applied models, it is also interesting to study computation time required by each of the models to achieve convergence of the network weights, as shown in Appendix C, Tables C.1 to C.6. For the WH network it is shown that computation time is higher for the sigmoidal activation function used in the output layer of the model, shown to be 321.81 seconds, compared to the linear summation activation function, which is shown to be 245.90 seconds. A study of computation time for the BP1 network has shown that increasing the number of hidden and input layer nodes increases computation time significantly. For instance, 12 input and 2 hidden layer nodes yields a computation time of 656.18 seconds, while the same number of inputs with 20 hidden layer nodes increases computation time to 3,361.77 seconds. Similar to computation time behaviour with changing network architecture for the WH and BP1 neural networks, increasing the number of hidden and input layer nodes yields an increase in computation time in the BP2 network. Computation time associated with 2 nodes in either hidden layer of the BP2 network is shown to be 1,068.51 seconds, increasing significantly to 4,263.03 seconds for 10 nodes in either hidden layer. Similarly, computation time is shown to increase in the RBF network for an increasing number of hidden layer nodes. For σ equal to 0.9, it is shown that computation time is 288.64 seconds for 5 hidden layer nodes, increasing substantially for 100 hidden layer nodes to 7,587.91 seconds. While the RBFKOH model has shown improved accuracy from the RBF model, it is important to note that computation time is significantly increased using the RBFKOH network, attributed to the computation time required for the Kohonen network to cluster the training data patterns. It is shown for the RBFKOH network with 20 hidden layer nodes and σ equal to 0.9 that 1,496.62 seconds is the time required to cluster the training data using the Kohonen network, where 1,000 iterations are required for weight convergence, while further processing time of 1,459.33 seconds is required to train the RBFKOH network. This combines to give a total computation time of 2,955.95 seconds, which is significantly higher than the computation time of 1,457.08 seconds required for the RBF network with 20 hidden layer nodes and σ equal to 0.9. Computation time for the GRNN is relatively low compared to the other

studied neural network models, attributed to the fact that this particular network requires only a single iteration for network training. It is shown for 10 pattern layer nodes and σ equal to 0.1 that computation time is 1.65 seconds, increasing to 77.29 seconds for 1,365 pattern layer nodes and the same σ value.

Applying the predictive importance technique to determine the influence of the input parameters on the output variables has shown that each of the applied neural networks use different features of the training data to make predictions of electrolyte additive quantity. The predictive importance analysis results achieved with each of the studied models for this application are documented in Appendix C, Tables C.7 to C.12, inclusive. It is important to note that a statistical *t*-test is used to compare the test data set error associated with the removal of each input parameter from the neural network with the original test data set error when all inputs are used. This is necessary to determine whether the observed increase or decrease in error corresponding to the removal of an input variable is statistically significant. A *t*-statistic value is documented in the results for each input parameter. The null hypothesis for the *t*-test is that the two population means are equal, while the alternative hypothesis is that the means are statistically significantly different. The null hypothesis is accepted if the *t*-statistic is lower in magnitude than the *t*-critical value, determined to be 2.345 in this instance for a significance level of 0.01 and a degree of freedom value of 199, else, the alternative hypothesis is accepted. For the WH model it is shown that target bath temperature, bath temperature, bath resistivity, AlF_3 and Na_2CO_3 additions (*t*-1), Na content and temperature reference are the only input variables contributing towards predictions, while the remaining inputs are not significant. Moreover, when the non-contributing inputs are removed from the network model and the WH network is re-trained and tested the RMS error improves slightly to 0.1073 and 0.1121 for the train and test data sets, respectively. On the other hand, it is shown that emf, AlF_3 addition (*t*-1) and Na content are not required as model variables for the BP1 network. When these particular variables are individually removed from the network model RMS error is shown to either decrease or not change in each instance from that when all network inputs are used. Moreover, removing these non-contributing variables collectively from the network model has shown an improved RMS error of 0.0713 and 0.0774 for the train and test data sets, respectively. The predictive and casual importance analyses for the BP2 network

have shown that all the specified network input variables are valued as contributing parameters for this application. It is shown that RMS error increases when any of the input variables are individually omitted from the network model. However, for the RBF network it is shown that emf and Na_2CO_3 addition (t-1) are non-contributing inputs for this application. Hence, the non-contributing inputs are removed from the network model, although no statistically significant improvement in RMS error is achieved. For the RBFKOH network it is shown that target bath temperature, bath temperature, bath resistivity, AlF3 addition (t-1), cell power, cell age, F content and temperature reference are contributing inputs to the network prediction. Removing the non-contributing inputs of bath height, emf, Na_2CO_3 addition (t-1) and Na content from the RBFKOH model has shown no statistically significant improvement in error. However, an improvement in the RMS error obtained with the GRNN using all inputs is achieved by removing non-contributing input variables from the model. It is shown that of the potential network inputs only target bath temperature, bath temperature, bath resistivity, AlF_3 addition (t-1) and temperature reference are contributing to the network prediction. Hence, the remaining input variables are removed from the model and the GRNN re-trained, producing a statistically significant improved RMS error of 0.0548 and 0.0671 for the train and test data sets, respectively.

Completing a casual importance analysis, with non-contributing input parameters removed as appropriate with regard to the predictive importance analyses, has confirmed the significance of the contributing input parameters for each of the studied neural network models. The results of the casual importance analysis completed in each instance for this particular application, as documented in Appendix C, Tables C.13 to C.18, inclusive, for each of the applied networks, has shown similar results to the corresponding predictive importance analysis. A statistical t-test is used to compare the test data set error associated with each varied input with the original test data set error. This is necessary to quantify the observed increase in error when contributing input parameters are varied in the data sets. For all contributing input variables, for each of the neural networks, it is shown that RMS error behaviour due to the variation of a particular input variable is comparable to that obtained when the same variable is omitted from the model. This is further highlighted by using the results obtained from the predictive and casual importance

techniques to calculate the percentage contribution of each input variable towards the network prediction. This is completed in order to rank input variable importance for each of the studied neural network models. As previously noted, percentage contribution of the input variables is calculated by summing the values of the error increase when an input variable is removed from the train and test data sets and dividing the error change for each variable by the sum. This yields a percentage contribution of each input variable to the total error. Appendix C, Tables C.19 and C.20 show the percentage contribution of the input variables using the predictive and casual importance techniques, respectively, for the applied neural network models. It can be seen that the percentage contribution of the input parameters using either technique is very similar in each instance. It is interesting to note that of the specified input variables, target bath temperature, bath temperature, bath resistivity and temperature reference were shown to contribute towards the network predictions for each of the neural networks applied to this industrial application. It is shown that while the percentage contribution of these input parameters is different for each of the neural network models, each of these variables have some influence on the prediction of AlF_3 and Na_2CO_3 quantity. This common behaviour indicates that target bath temperature, bath temperature and bath resistivity to have a significant correlation with electrolyte chemistry. Temperature reference, while not a parameter of the Hall-Heroult process, is shown to be necessary for the neural network modelling, as it is shown to specify the time delay between a chemical addition to the electrolyte and the previous electrolyte temperature measurement. Hence, a useful result of the neural network modelling completed thus far is that it has been beneficial for improving the understanding of relationships among critical parameters of the Hall-Heroult process. The knowledge of these relationships is crucial in developing and improving process control methodologies.

While the network models applied to this industrial application have been assessed on an individual basis in the preceding documentation, it is useful to summarise the results of the neural network modelling for this particular application with a comparison of the studied neural network models. In particular, it is interesting to compare the minimum RMS error and computation time achieved by each of the applied models, as shown in Table 5.2.4. It is important to note that the RMS error values shown in the table correspond to the lowest RMS error achieved with each

network by optimising the architecture and removing non-contributing inputs from the model. Further, the computation time shown is that associated with the network architecture and number of input parameters that produced the lowest RMS error.

TABLE 5.2.4. Comparison of Neural Network Modelling Results for Electrolyte Additive Prediction Application

	Neural Network Model					
Property	WH	BP1	BP2	RBF	RBFKOH	GRNN
RMS error - train	0.1073	0.0713	0.0615	0.0634	0.0606	0.0548
RMS error - test	0.1121	0.0774	0.0719	0.0727	0.0715	0.0671
comp. time (s)	281.0	1,010.3	3,089.1	1,319.4	2,711.2	24.6

It can be seen from the neural network modelling results that RMS error and computation time are different for each of the applied models and further, that the influence of the input variables on the network prediction in each instance is significantly different. It can be seen that while some networks have lower RMS error than others, a higher number of process variables are required to achieve the low error. For instance, considering the WH, BP1 and BP2 models, it is shown that while the addition of a hidden layer in the network from the WH to BP1 model yields a decreases in error, the number of input variables required increases from 7 to 9 for the WH and BP1 networks, respectively. Further, for the BP2 model, the addition of a second hidden layer decreases RMS error from that of the BP1 network, however, the number of required input variables increases to 12. On the other hand, while RMS error using the RBF and RBFKOH models has shown RMS error to be lower in the RBFKOH network, the number of input variables required in the RBFKOH network is also lower, shown to be 10 and 8 for the RBF and RBFKOH networks, respectively. Further, the GRNN has shown lowest RMS error of the applied neural network models, while the number of input variables required in the GRNN model is also the lowest of the applied networks. It is shown that the GRNN requires only 5 of the specified input variables for network predictions. It is interesting to note that the train and test RMS error achieved with each of the neural network models applied here is lower than the RMS error associated with the MVRA. Further, it is shown that the percentage contribution of the input variables obtained using the MVRA is different to that obtained for each of the applied neural networks. Nevertheless, the MVRA has exhibited significantly lower computation time than the applied neural

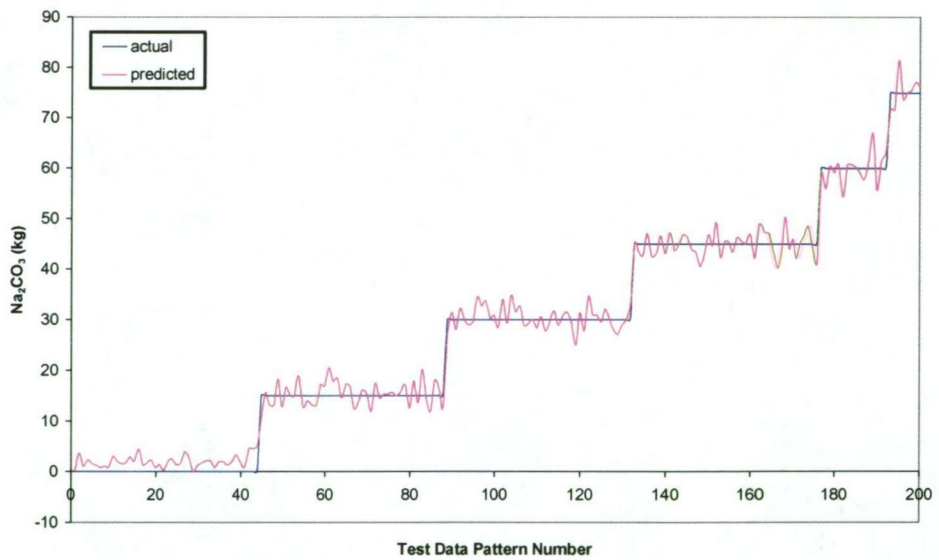
network models, attributed to the lower complexity of the regression technique compared to the applied neural networks.

From the neural network comparison given it is shown that there is an obvious correlation between computation time and network complexity. For instance, considering the WH, BP1 and BP2 networks, it is shown that while each of these networks train using the backpropagation procedure, the WH network, which has no hidden layer, has a relatively low computation time, the BP1 network has a higher computation time than the WH due to the addition of a hidden layer in the network architecture, and the BP2 network has the highest computation time of the three networks, attributed to the use of two hidden layers. Further, it is shown that while the Kohonen network for clustering similar training data patterns is useful for decreasing RMS error in the radial basis function network, adopting this clustering technique significantly increases computation time. The GRNN, while having a large number of nodes in the network architecture, is shown to have a very low computation time, attributed to the fact that there are no iterations required for convergence of the network weights in this model. It is shown that the weights in the GRNN are set in a single pass of the training data patterns.

While a comparison of RMS error, percentage contribution of the input variables and computation time is completed for each of the applied neural networks, it is useful to study the practical implications of applying neural networks to this industrial application. This is completed by considering the actual and predicted values of electrolyte additive quantity. Moreover, it is useful to consider the particular neural network that produced lowest RMS error in this instance, shown to be the GRNN model. A useful demonstration of this is a plot of actual and predicted values of AlF_3 and Na_2CO_3 for the test data set, as shown in Figure 5.2.8, in which the test data patterns have been sorted from lowest to highest value to improve the readability of the graph. Further, it is useful to note that the actual addition of AlF_3 to the electrolyte is in 10.5kg quantities, with 0.0kg being the lowest addition and 73.5kg being the highest addition, while Na_2CO_3 is added in 15.0kg quantities, with 0.0kg being the lowest addition and 75.0kg being the highest addition. Hence, the stepwise behaviour of the graphs represents the different addition quantities for these two electrolyte additives. It can be seen that the predicted values of AlF_3 and Na_2CO_3

follow quite closely the actual values of AlF_3 and Na_2CO_3, respectively, in each instance.

(a)

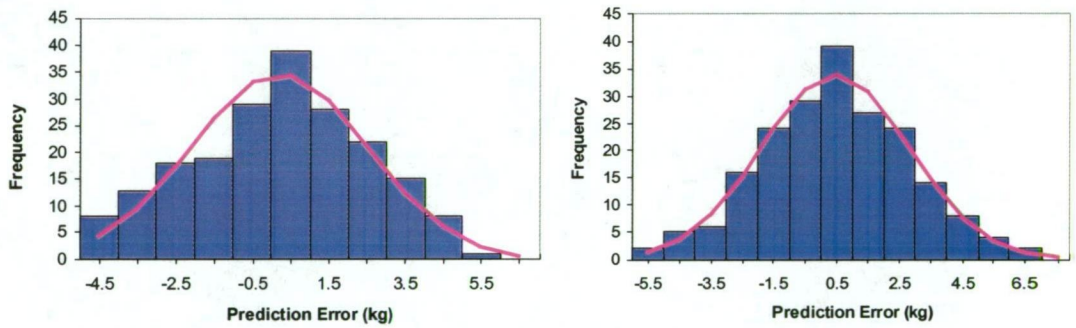


(b)

Fig. 5.2.8. Comparison of Actual and GRNN Predicted Values for Electrolyte Additive Prediction Application. (a) AlF_3, and (b) Na_2CO_3

It is useful to consider the distribution of prediction error associated with the GRNN, as shown in Figure 5.2.9(a) and (b) for the test data set for AlF_3 and Na_2CO_3, respectively. Considering firstly the error associated with AlF_3 predictions, it can be seen that the majority of error is very close to 0.0kg. Further, the error distribution

approximates a normal distribution, as highlighted by the normal curve approximation of the histogram, slightly skewed positive, which is attributed mostly to the error corresponding to predictions associated with an actual AlF_3 and Na_2CO_3 content of 0.0kg. It is important to note that predictions associated with an actual AlF_3 and Na_2CO_3 content of 0.0kg cannot be negative, hence, all error in this region is positive. On the other hand, for actual additions of AlF_3 and Na_2CO_3 greater than 0.0kg, predictions can be lower-than-actual, therefore, negative error. Hence, while there is an even distribution of higher-than-actual and lower-than-actual predictions for actual AlF_3 and Na_2CO_3 additions greater than 0.0kg, as demonstrated in Figure 5.2.8, only zero error or higher-than-actual predictions, which yield positive error, are possible for actual AlF_3 and Na_2CO_3 additions of 0.0kg. Hence, this contribution of positive error is responsible for the slightly skewed behaviour exhibited by the normal curve approximation of the histogram.



(a) (b)

Fig. 5.2.9. Histogram Showing Prediction Error Distribution for GRNN for Electrolyte Additive Prediction Application. (a) AlF_3, and (b) Na_2CO_3

In addition, it is interesting to consider the histogram statistics, as presented in Table 5.2.5. The slightly positive skew of the histogram in either instance is confirmed by the skewness values of 0.18 and 0.22 for AlF_3 and Na_2CO_3, respectively. It is useful to note that the skewness value specifies the degree of symmetry of a distribution about the mean. A positive value indicates the data is skewed to the right of the mean, while a negative value indicates that the data is skewed left. Moreover, a skewness value close to zero indicates that the distribution is symmetrical about the mean. In addition, it can be seen that the maximum prediction error has a magnitude of 5.62 and 6.90kg for AlF_3 and Na_2CO_3, respectively. Further, the value of the 50^{th}

percentile, or mean, is shown to be 0.21 and 0.48kg for AlF_3 and Na_2CO_3, respectively. Further, it is shown that approximately 46.0% of prediction error is lower than 0.0kg for AlF_3 and approximately 42.0% of prediction error is lower than 0.0kg for Na_2CO_3.

TABLE 5.2.5. Prediction Error Histogram Statistics for GRNN for Electrolyte Additive Prediction Application

Descriptive Statistics	AlF_3	Na_2CO_3
data size (test)	200	200
minimum value (kg)	-4.88	-5.70
maximum value (kg)	5.62	6.90
range (kg)	10.50	12.60
mean value, or 50^{th} percentile (kg)	0.21	0.48
standard deviation (kg)	2.30	2.36
skewness (kg)	0.18	0.22
proportion of error below 0.0kg	0.46	0.42

The average prediction error associated with the specified electrolyte additive quantity for AlF_3 and Na_2CO_3 is shown in Table 5.2.6. It is important to note that this error is an average absolute error. That is, only the magnitude of the difference between predicted and actual electrolyte addition quantity is considered, not whether the prediction is higher- or lower-than-actual. This is necessary to consider in regard to practical implementation of the neural network as it is the magnitude of the difference between actual and predicted values that is important. Decreasing process variation is significantly influenced by the ability of the neural network to achieve highly accurate predictions, not whether the neural network can on average have an equal number of higher- and lower-than-actual predictions of equivalent magnitude. In regard to AlF_3, it can be seen that there is low error associated with the majority of addition quantities, with lower accuracy shown in the higher addition quantities. This is attributed to the fact that higher addition quantities were not as well represented in the training data patterns due to the rare occurrence of additions in this extreme of the operating range. Likewise, while Na_2CO_3 predictions have also shown high accuracy, error increases for high addition quantities, also attributed to insufficient training data at the extreme of the operating range. The percentage of additions for each of the additive quantities is also shown in the table to highlight the frequency at which AlF_3 and Na_2CO_3 are administered to the reduction cell. It is shown for AlF_3 that 0.0 and 10.5kg are the most popular addition quantities, while popularity decreases with

increasing quantity. Similarly, for Na_2CO_3 popularity decreases with increasing quantity, while 0.0kg is shown to be the most common addition of this important electrolyte additive.

TABLE 5.2.6. Prediction Error Associated with Specified Data Ranges for Electrolyte Additive Prediction Application

Electrolyte Additive Quantity (kg)	Average Prediction Error for Quantity (kg)	Percentage of Additions for this Quantity (%)
	Aluminium Fluoride - AlF_3	
0.0	1.77	21.4
10.5	1.78	20.6
21.0	1.76	18.5
31.5	1.85	17.7
42.0	1.90	12.5
52.5	2.17	7.8
63.0	2.96	1.1
73.5	3.41	0.4
	Soda-Ash - Na_2CO_3	
0.0	1.75	61.4
15.0	1.80	17.3
30.0	2.05	10.8
45.0	2.19	8.4
60.0	2.29	1.4
75.0	2.53	0.7

For the majority of AlF_3 and Na_2CO_3 addition quantities it is shown that the neural network modelling in this instance has achieved a high level of accuracy. This level of accuracy gives confidence in implementing a neural network to schedule electrolyte additives to the reduction cell at CABBL to investigate the resulting effect on reduction cell behaviour. In particular, it has been stated in a previous chapter that the objective of this prediction methodology is to minimise electrolyte temperature variation in the reduction cell, leading to improved current efficiency and consequently, reduced processing costs. Hence, performance assessment for neural network modelling for this application will be through a study of electrolyte temperature variation in the reduction cell, with particular attention to electrolyte temperature standard deviation. It has been shown that electrolyte temperature is predominantly controlled by the addition of AlF_3 and Na_2CO_3 to the reduction cell. Hence, improving the accuracy of the required quantity of electrolyte additive needed to return the electrolyte temperature to an optimum level, shown previously to be

965.0°C at CABBL, will have significant influence on electrolyte temperature variation. However, it is not possible to determine the success of this electrolyte additive prediction methodology from the neural network training and testing results alone. While the neural network training and testing completed indicates that a suitable process model is developed in this instance, it is necessary to implement the neural network 'on-line' to study the resulting effect on electrolyte temperature variation. The results of this implementation are detailed in a following chapter.

However, while the GRNN model is shown to have the lowest associated RMS error of the applied neural networks in this instance, it is important to note that it is not necessarily the most economically suitable process model for this particular industrial application. In addition to minimising prediction error associated with the process model, it is important to consider the economic cost associated with data acquisition of the associated model variables. As a result of neural network modelling for this application it is shown that the number of input variables required for each model is different and further, the percentage contribution of the input variables for each model is significantly different. When selecting a neural network model for a particular application it is important to consider the most economically suitable model, which is not necessarily the model that yields highest accuracy. In some instances it may be economically beneficial to adopt a network model that has higher associated prediction error but requires a lower number of process parameters. Moreover, it may be economically feasible to compromise prediction accuracy for a lower number of process variables required, if the cost of acquiring process data for specific parameters is higher than the cost associated with prediction accuracy. While a strategy for selecting a neural network based on economic consideration is introduced and discussed in a later chapter, it is useful to note here that a selection methodology is required due to the complexity associated with the decision of which neural network model to use for a specific application. Considering the results of the neural networks applied to the following industrial application further highlights the complexity associated with this decision.

5.3 ASSESSMENT OF NEURAL NETWORK MODELS FOR CELL FAILURE PREDICTION

The 24 process parameters selected as potential neural network inputs for this cell failure prediction application are shown in Figure 5.3.1, using an arbitrary neural network architecture, together with the single network output, in order to highlight the particular input-to-output mapping in this instance.

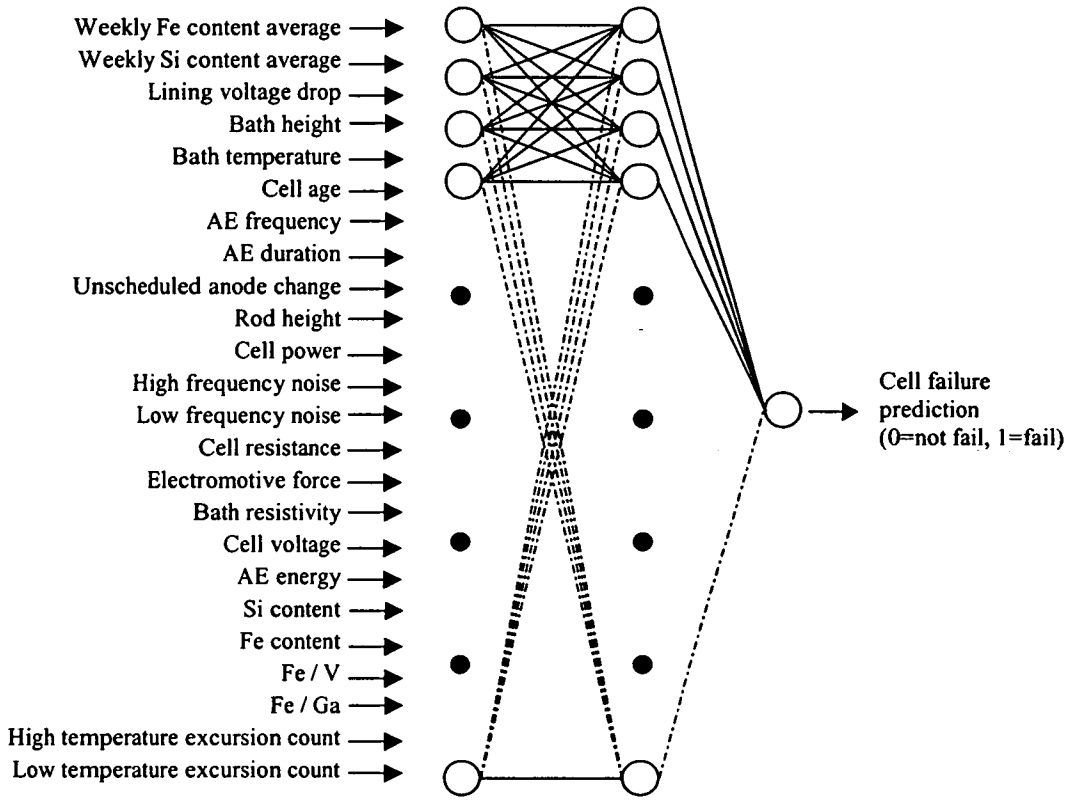
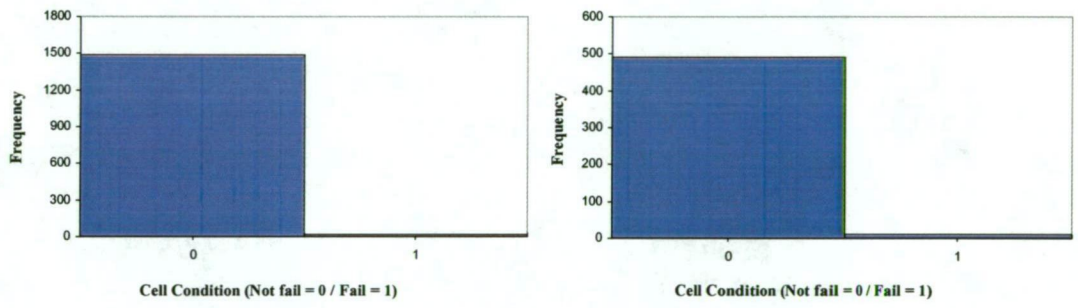


Fig. 5.3.1. Illustration of Neural Network Model for Cell Failure Prediction Application

Similar to the previous application, data acquisition was completed by retrieving suitable information from the smelter knowledge base. Data formatting and pre-processing, which involved the elimination of corrupt information from the acquired data and ensuring sufficient information for each model variable was included in the data, produced 2,000 data patterns in total for this particular application. The processed data was then divided into 1,500 training data patterns and 500 test data patterns. Distribution of the output variable in the training and test data sets is shown in Figure 5.3.2, from which it can be seen that representation of the not-fail condition is significantly higher than the fail condition. In particular, in the training data set

1,484 example patterns representing a not-fail condition are given compared to 16 fail condition example patterns, while in the test data set 490 not-fail condition example patterns are given compared to 10 example patterns representing a fail condition. However, there is a justification for the distribution of the output variable value in the training and test data sets being significantly biased towards the representation of a not-fail condition. It is limited by the number of cell failures that occurred during the period for which suitable neural network training and test information could be retrieved from the smelter knowledge base. While there was substantial information available for a not-fail condition, a total of only 26 cell failures had occurred during the suitable period, hence, limiting the number of training and test examples representing a cell fail condition to 26, divided into 16 and 10 for the train and test data sets, respectively.



(a) (b)

Fig. 5.3.2. Histogram Showing Distribution of the Network Output Variable in the Data Sets. (a) Training, and (b) Test.

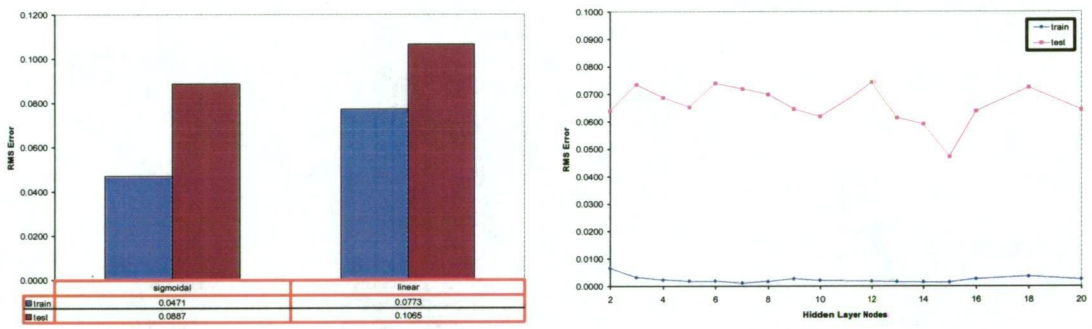
The results of the MVRA completed for this application are shown in Table 5.3.1. For a sample size of 1,500, 24 source variables and a confidence level of 95.0%, a *t*-critical value of 1.645 is appropriate for the statistical *t*-test in order to assess the significance of the source variables in the regression model. Hence, the magnitude of the *t*-statistic is required to be higher than 1.645 for a regression coefficient to be significantly different to zero. The statistical significance of each source variable is noted in the table. It is shown that an *adjusted* r^2 value of 0.2715 is achieved in this instance using the specified regression model, highlighting that while approximately 27.1% of the variation of the output variable is explained by variation of the input variables, 72.9% of the output variable variation is unexplained by the regression model. Further, it is shown that of the available source variables only Fe weekly,

lining voltage drop, cell age, unscheduled anode change, low frequency noise, Fe, Fe/V and high temperature excursion count are significant contributing parameters in the regression model. The percentage contribution of the contributing variables for the regression model is determined using the previously discussed technique. It is shown that Fe/V (38.8%) is the most significant of the source variables in the MVRA model, followed by Fe weekly (32.1%), Fe (23.2%), high temperature count (1.7%), low frequency noise (1.5%), lining voltage drop (1.0%), unscheduled anode change (0.9%) and cell age (0.8%). In addition, comparing the actual cell failure events with the MVRA predictions has shown a RMS error for the train and test data sets of 0.2871 and 0.3264, respectively. This information is useful for later comparison with the applied neural network models.

TABLE 5.3.1. Multi-Variable Regression Analysis Results for Cell Failure Prediction Application

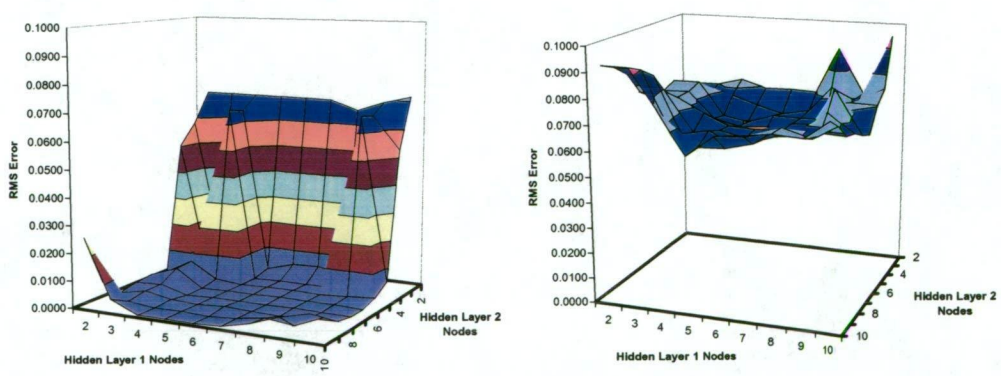
Regression Statistics			
r: 0.5284	r^2: 0.2793	*adjusted* r^2: 0.2715	*observations*: 1,500
Parameter	Regression Coefficient	t-statistic	Significant
Fe content (weekly)	1.85293	6.63420	yes
Si content (weekly)	0.00184	-0.96517	no
lining voltage drop	-0.05596	-1.70658	yes
bath height	-0.00647	1.20367	no
bath temperature	0.03219	0.97804	no
cell age	-0.04896	-1.70977	yes
AE frequency	0.04154	1.48408	no
AE duration	0.01919	0.64830	no
unsched. anode change	-0.05164	-3.39818	yes
rod height	0.02810	0.79426	no
cell power	0.01082	0.38266	no
high frequency noise	0.01984	0.46433	no
low frequency noise	-0.08790	-3.16861	yes
cell resistance	-0.00992	-0.41424	no
emf	0.01101	1.03651	no
bath resistivity	-0.00687	1.05870	no
cell voltage	-0.00324	-0.95324	no
AE energy	-0.00693	-0.34571	no
Si content	0.00417	0.35741	no
Fe content	-1.33525	-6.34827	yes
Fe/V	2.23555	8.45823	yes
Fe/Ga	0.01252	-0.98521	no
high temperature count	0.09598	4.74744	yes
low temperature count	0.01421	1.23400	no

For each of the studied neural networks for this application the behaviour of RMS error with changing network architecture is highlighted in Figures 5.3.3 to 5.3.7, inclusive, and tabulated in Appendix C, Tables C.21 to C.26, inclusive. While the practical implications of the RMS error values associated with the applied neural networks is assessed later in this section in regard to cell failure prediction, it is useful to note that the applied neural networks have shown high accuracy in this instance. In addition, it is interesting to note that similar to the previous application, the neural networks have exhibited changing RMS error with changing neural network architecture and algorithm. This behaviour confirms the need to study RMS error with changing architecture and algorithm in order to achieve minimum prediction error.



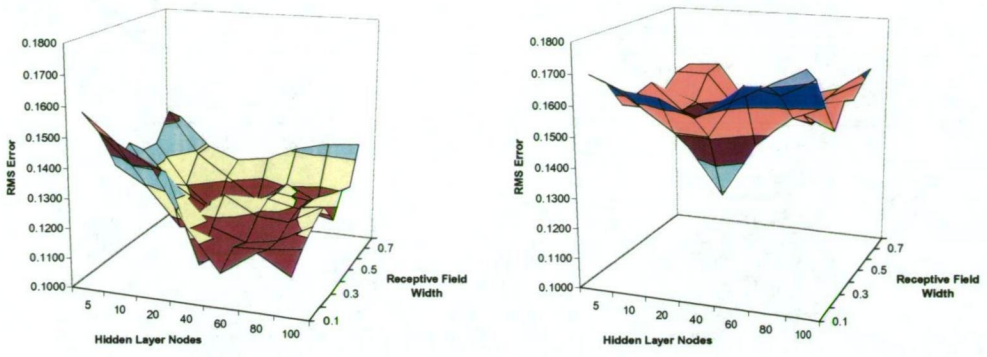
(a) (b)

Fig. 5.3.3. RMS Error Behaviour with Changing Architecture. (a) WH Network, and (b) BP1 Network



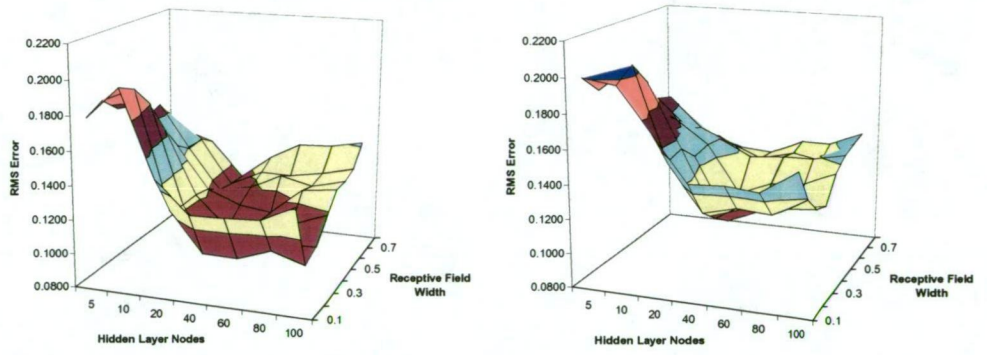
(a) (b)

Fig. 5.3.4. BP2 Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



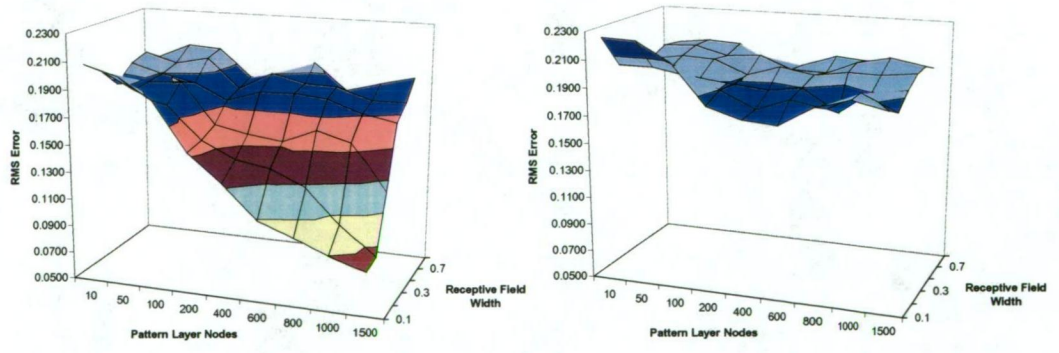
(a) (b)

Fig. 5.3.5. RBF Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.3.6. RBFKOH Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.3.7. GRNN RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

The WH neural network applied to this particular industrial application has shown the sigmoidal activation function in the output layer nodes produces RMS error

values of 0.0471 and 0.0773 for the train and test data sets, respectively. It is shown that this is lower than the RMS error of 0.0887 and 0.1065 for the train and test data sets, respectively, achieved using the linear summation activation function in the output layer nodes. A study of RMS error behaviour with increasing iterations, to maximum of 1,000 iterations, has shown weight convergence in the WH network occurs after approximately 200 iterations. However, the BP1 network has shown an improvement in RMS error from the WH model. It is shown that a RMS error of 0.0014 and 0.0472 is achieved using 15 hidden layer nodes in the BP1 network. It is important to note that this was achieved using the sigmoidal activation function in the hidden and output layers of the network, which has shown, while not documented here, lower error than the linear summation function in the output layer nodes. The error achieved with the BP1 network is obtained after approximately 650 iterations, after which RMS error does not decrease, highlighting convergence of the network weights. Considering the BP2 network, it is shown that 6 nodes in the first hidden layer and 5 nodes in the second hidden layer of the network produce the minimum error for this application. It is shown that using this architecture with the sigmoidal activation function in the hidden and output layers of the network, a RMS error of 0.0007 and 0.0581 for the train and test data sets, respectively, is achieved. Further, approximately 800 iterations are sufficient for network weight convergence to occur in this instance. The RBF model has shown higher error than the WH, BP1 and BP2 networks in this instance. It is shown that the minimum RMS error achieved with the RBF network is 0.1178 and 0.1287 for the train and test data sets, respectively. It is shown that this is obtained using 40 hidden layer nodes and a σ value of 0.3. Further, a study of RMS error over a duration of 1,000 iterations has shown weight convergence to occur after 250 iterations. Clustering similar training data patterns and using these clustered patterns as network weights, as completed for the RBFKOH network, has shown an improvement in RMS error compared to the RBF model. It is shown that a RMS error of 0.1043 and 0.1081 for the train and test data sets, respectively, is the minimum error that can be achieved with the RBFKOH network. It is shown that this is obtained using 40 hidden layer nodes and a σ value of 0.4. Similar to the RBF network, 250 iterations are sufficient for weight convergence in the RBFKOH model, determined from a study of RMS error behaviour over a duration of 1,000 iterations. The GRNN model in this instance has shown the highest RMS error of any of the studied neural network models for this

industrial application. It is shown that the lowest RMS error achievable with the GRNN model was 0.1326 and 0.1620 for the train and test data sets, respectively, obtained using 400 pattern layer nodes and a σ value of 0.1.

The statistical significance of the difference in error associated with each model is evaluated using the statistical ANOVA technique. Similar to the previous application, there are 7 populations in this analysis, representing the error associated with each model, and the test data set error is used, which has been shown to have 500 data patterns. Hence, for ν_1 equal to 6, ν_2 equal to 3,493 and α equal to 0.05 a critical test statistic value of 2.1012 is appropriate. The results of the ANOVA study for this application are documented in Table 5.3.2. The null hypothesis for this investigation is that the population means are equal, while the alternative hypothesis is that at least two population means are unequal.

TABLE 5.3.2. Test Data Set Error ANOVA Results for Cell Failure Prediction Application

ANOVA Statistics			
MST: 0.0377	*MSE*: 0.0138	*F*: 2.7220	F_α: 2.1012
Population	Mean	Variance	
p_1 - (WH)	-0.0053	0.0050	
p_2 - (BP1)	0.0056	0.0234	
p_3 - (BP2)	0.0082	0.0065	
p_4 - (RBF)	0.0065	0.0232	
p_5 - (RBFKOH)	-0.0062	0.0227	
p_6 - (GRNN)	-0.0084	0.0247	
p_7 - (MVRA)	0.0170	0.0475	

It is shown that the null hypothesis is rejected as the magnitude of the test statistic exceeds the critical value. It is shown that the test statistic in this instance is 2.7220, which is greater than the critical value of 2.1012. Hence, it is confirmed that at least two populations means are statistically significantly different, requiring a series of statistical *t*-tests to determine which population means are different. The results of these *t*-tests are documented in Table 5.3.3. A *t*-critical value of 2.333 is obtained for α equal to 0.01, to minimise the probability of a Type I error occurring, and a degree of freedom value of 499. It is shown that the *t*-statistic value for each paired population is greater than the *t*-critical value in each instance. Therefore, the null hypothesis is rejected for all paired populations, hence, all population means are

statistically significantly different. This confirms that the error associated with any one of the applied models for this application is statistically significantly different to the error associated with any other applied model. Hence, no two models exhibit statistically equivalent error.

TABLE 5.3.3. Test Data Set Error Statistical *t*-Test Results for Cell Failure Prediction Application

t_1 to t_7	t_2 to t_7	t_3 to t_7	t_4 to t_7	t_5 to t_7	t_6 to t_7
$t_{12} = 3.032$	$t_{23} = 6.826$	$t_{34} = 6.637$	$t_{45} = 8.140$	$t_{56} = 3.804$	$t_{67} = 4.007$
$t_{13} = 5.956$	$t_{24} = 2.964$	$t_{35} = 3.519$	$t_{46} = 5.273$	$t_{57} = 4.194$	
$t_{14} = 3.329$	$t_{25} = 4.876$	$t_{36} = 2.407$	$t_{47} = 2.688$		
$t_{15} = 5.592$	$t_{26} = 3.204$	$t_{37} = 5.634$			
$t_{16} = 4.793$	$t_{27} = 2.736$				
$t_{17} = 3.190$					

Considering computation time for each of the applied neural networks and the appropriate number of iterations completed in each instance, as documented in Appendix C, Tables C.21 to C.26, inclusive, it is shown that there is a broad range of computation times required by the different neural networks to develop a suitable process model. Moreover, a study of computation time for the WH network has confirmed previous findings that the sigmoidal activation function used in the output layer of the network gives rise to a higher computation time than the linear summation activation function. It is shown that computation time is 311.82 seconds for the sigmoidal function, decreasing to 246.67 seconds for the linear summation function. Further, computation time is shown to increase significantly with increasing number of hidden layer nodes in the BP1 network. It is shown that computation time is 677.42 seconds for 2 hidden layer nodes, increasing to 5,951.32 seconds for 20 hidden layer nodes. This highlights the importance of using the minimum number of hidden layer nodes possible to achieve minimum RMS error, in order to minimise computation time. For the BP2 network, computation time is shown to be significantly higher than the WH and BP1 models, shown to be 3,068.63 seconds for the BP2 architecture producing minimum error, shown to be 6 and 5 nodes in the hidden and output layers, respectively. This is attributed to the higher complexity of the BP2 network, due to the use of 2 hidden layers in the network architecture. For the RBF network, while σ has been shown to have no affect on computation time, it is shown that the number of hidden layer nodes influences computation time

significantly. It is shown, for σ equal to 0.3, that for 5 hidden layer nodes, computation time is 275.02 seconds, increasing to 5,195.19 seconds for 100 hidden layer nodes. Due to the computation time associated with clustering the training patterns using the Kohonen network, computation time for the RBFKOH network is shown to be significantly higher compared to the RBF model. It is shown that total computation time for the RBFKOH network with 40 hidden layer nodes and a σ value of 0.4 is 5,157.87 seconds, comprised of 3,043.35 seconds for Kohonen clustering, using 1,000 iterations, and 2,114.52 seconds for further RBFKOH training. Computation time for the GRNN is shown to be significantly lower than that of the other studied neural network models, which is due to the fact that this particular network requires only a single iteration for network training. It is shown, for a σ value of 0.1, that for 10 pattern layer nodes computation time is 1.65 seconds, increasing to 122.10 seconds for 1,500 pattern layer nodes and the same σ value.

The results of the predictive importance analyses for the studied neural networks are documented in Appendix C, Table C.27 to C.32, inclusive, for this particular application, with an appropriate *t*-statistic value noted for each input parameter. The *t*-critical value in this instance is 2.333, for a significance level of 0.01 and a degree of freedom value of 499. Similar to the previous application, the test data set error is used for the comparison of population means. For the WH model it is shown that the majority of the specified input variables contribute towards network predictions. It is shown that bath height, emf, low frequency noise and Fe/Ga are non-contributing inputs in the WH model. Hence, these inputs are removed from the train and test data patterns, resulting in a statistically significant improvement in error, shown to be 0.0470 and 0.0761 for the train and test data sets, respectively. For the BP1 model it is shown that low temperature excursion count is the only non-contributing input variable in the model, thus, the remainder of the input parameters are required for making predictions. Removing low temperature excursion count from the BP1 model yields an RMS error of 0.0021 and 0.0451 for the train and test data sets, respectively, which is a statistically significant improvement in error for the test data set. For the BP2 network it is shown that bath height and emf are not contributing towards network predictions. Although, re-training the BP2 network with these non-contributing variables removed has shown no improvement in error. An analysis of input importance for the RBF network has shown bath height, emf, bath resistivity,

cell voltage, Si content and low temperature excursion count are not used in the RBF model for predictions of cell failure. Hence, these non-contributing parameters are removed from the network model to achieve an improved RMS error of 0.1156 and 0.1279 for the train and test data sets, respectively. While a lower RMS error is obtained with the RBFKOH network, compared to the RBF model, it is shown also that a lower number of input variables are required in the RBFKOH network. It is shown that Si weekly content average, bath height, AE duration, cell resistance, emf, bath resistivity, cell voltage, Si content, Fe/Ga and low temperature excursion count are non-contributing network inputs, while the remainder are shown to be statistically significant for predictions of electrolyte temperature. Omitting the non-contributing inputs from the RBFKOH model and re-training has resulted in a slightly improved RMS error of 0.1031 and 0.1074 for the train and test data sets, respectively. Further, the predictive importance analysis for the GRNN has shown bath temperature, cell age, AE frequency, AE duration, unscheduled anode change, rod height, cell power, cell resistance, emf, bath resistivity, AE energy and Fe/Ga are non-contributing network inputs. Removing these non-contributing variables from the GRNN model has shown a statistically significant decrease in RMS error to 0.1319 and 0.1598 for the train and test data sets, respectively.

Similar to the previous application, RMS error behaviour exhibited by each of the studied neural networks for all contributing model parameters during the casual importance analysis is the same as that observed using the predictive importance technique. Further, a statistical analysis has confirmed the contributing input parameters associated with each model yield an increase in error when varied in the model. The casual importance results are documented for this application in Appendix C, Tables C.33 to C.38, inclusive, for each of the applied models. In addition, Appendix C, Tables C.39 and C.40 document the percentage contribution of the input variables using the predictive and casual importance analysis results, respectively. It is shown that the percentage contribution of the input variables is comparable in either instance. While the percentage contribution of Fe weekly content average, lining voltage drop, high frequency noise, Fe content, Fe/V and high temperature excursion count is shown to be different for each the applied neural network models, these particular process parameters are shown to have some influence in each network model for the prediction of cell failure. This is a useful

result to emanate from the neural network modelling completed for this application as it improves the understanding of the aluminium smelting process, which is critical knowledge for enhancing process control.

While a discussion of the individual neural network models is completed, a comparison of minimum RMS error and computation time achieved with each of the applied network models is useful, as shown in Table 5.3.4. Similar to the previous application, the RMS error shown in the table corresponds to the lowest RMS error achieved with each network by optimising the architecture and removing non-contributing inputs from the model. Likewise, the computation time shown is that associated with the network architecture and number of inputs that produced the lowest RMS error.

TABLE 5.3.4. Comparison of Neural Network Modelling Results for Cell Failure Prediction Application

	Neural Network Model					
Property	WH	BP1	BP2	RBF	RBFKOH	GRNN
RMS error - train	0.0470	0.0021	0.0007	0.1156	0.1031	0.1319
RMS error - test	0.0761	0.0451	0.0580	0.1279	0.1074	0.1598
comp. time (s)	271.3	4,438.2	2,871.1	1,793.3	4,547.4	24.8

It can be seen from a comparison of the neural network modelling results that the RMS error, computation time and ranking of importance of the input variables is significantly different for each of the applied networks. It is shown that the lowest RMS error is achieved using the BP1 model, which requires 23 of the specified input parameters, whereas the highest RMS error is obtained with the GRNN, but only 12 of the specified input variables are used. Considering the RBFKOH network, it is interesting to note that while RMS error decreases from that obtained with the RBF model, the required number of input variables also decreases. Further, while comparing the neural networks it is necessary to note that computation time for each of the networks in this instance have exhibited similar behaviour to that observed for the previous application. Specifically, the applied neural networks exhibit higher computation time for increasing network complexity and increasing number of iterations required for convergence of the network weights. Considering the RMS error associated with the MVRA, shown to be 0.2871 and 0.3264 for the train and

test data sets, respectively, it can be seen that a significantly lower RMS error was achieved with each of the applied neural network models. This is a similar result as that obtained for the previous application. Further, while the significance of the input parameters is shown to be different for the MVRA model compared to the applied neural networks, computation time is shown to be comparatively low for the regression model.

While a study of RMS error is useful to select the best network architecture for each model and study the features of the training data used during modelling, it is also useful to consider the practical implications of the neural network modelling completed in this instance. The results presented here are those achieved using the BP1 model, as this particular neural network has shown the lowest RMS error in this instance. While the output of the BP1 network is a value in the bounded range 0.0 to 1.0, due to the sigmoidal activation function used in the output layer of the network, for practical implementation of the developed BP1 neural network model it is necessary to round the output of the network to the nearest whole number, either 0 or 1. This is necessary to classify a particular input data pattern for a reduction cell as either not-fail, given by a 0 as the rounded predicted value, or fail condition, given by 1 as the rounded predicted value. Completing the rounding of the predicted output values and comparing the actual and predicted values for the test data set, as shown in Figure 5.3.8, yields a model with 100.0% accuracy. Although the graph appears to have a single line plotted it indeed has two lines, representing the actual and predicted values. It appears as a single line due to the high accuracy of the model, the actual and predicted values compare with zero error. It is shown for the 10 fail conditions distributed randomly throughout the test data set that the BP1 neural network model predicts the 10 fail conditions in each instance, represented by a 'spike' of value 1 on the graph. Further, it is shown that the BP1 model also predicts a not-fail condition with zero error, represented by a value of 0 on the graph. This is an important result as it highlights the ability of the network to not falsely predict a fail condition, eliminating unnecessary cell cut-out, while maintaining 100.0% accuracy for identifying cell failure conditions.

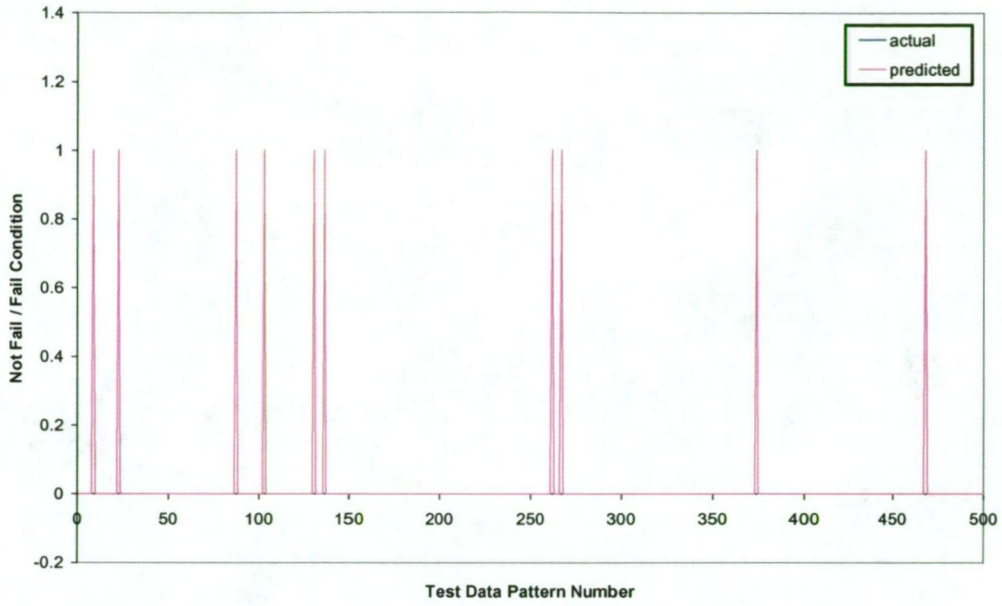


Fig. 5.3.8. Comparison of Actual and BP1 Neural Network Predicted Values for Cell Failure Prediction Application

While a distribution of prediction error is completed for the previous application it is not appropriate to do the same in this instance as it is shown that there is zero prediction error associated with the developed neural network model, ie. for the test data set population of 500 the distribution of error is a frequency of 500 of value 0.0. Further, the minimum, maximum and mean value and standard deviation of the error distribution all equal 0.0. Nevertheless, the high accuracy of the developed neural network model in this instance gives confidence in implementing a neural network on-line at CABBL to predict reduction cell failure, with the objective of significantly reducing or eliminating tap-out occurrences at the smelter while maximising cell life. The specific details and results of the neural network implementation for this particular application are documented in a later chapter.

However, it is useful to note that the neural network models applied to this industrial application have shown a significant difference in prediction accuracy and the number of process parameters used in the developed model. As introduced in the previous application and reinforced here by the diverse range of modelling results, it becomes necessary to have a suitable methodology for network selection in order to maximise the economic benefit from neural network modelling. It is useful to note here that the neural network modelling results obtained for this application will be

critical for investigating a particular neural network selection technique detailed in a following chapter. Another application is considered here to further highlight the need for a neural network selection strategy for making a decision on the most economically feasible model to implement for a specific application.

5.4 ASSESSMENT OF NEURAL NETWORK MODELS FOR ELECTROLYTE TEMPERATURE PREDICTION

It has been shown that there are 10 process parameters selected as potential neural network inputs for electrolyte temperature prediction and also that electrolyte temperature is the single output associated with this application. Using an arbitrary neural network architecture, the input-to-output mapping in this instance is highlighted in Figure 5.4.1.

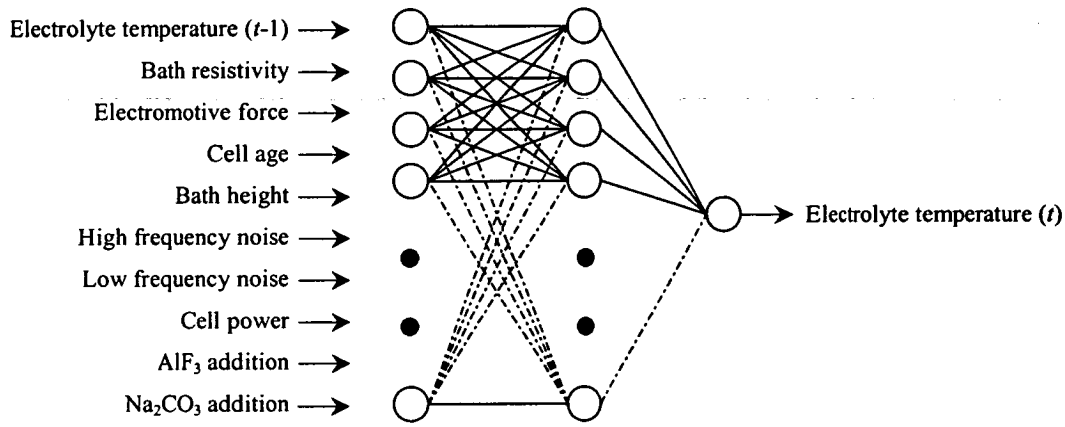
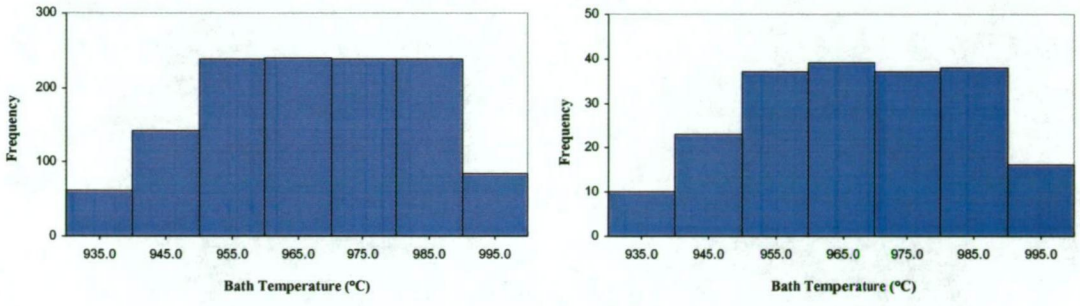


Fig. 5.4.1. Illustration of Neural Network Model for Electrolyte Temperature Prediction Application

A total of 1,444 data patterns were obtained from the smelter knowledge base for the specified input and output variables. Similar to the previously documented industrial applications, data preparation formed a complex stage of network modelling in this instance. The processed data was divided into 1,244 training data patterns, with the remaining 200 patterns used as test data. Figure 5.4.2 shows an approximately uniform distribution of data for the network output variable for the train and test data sets, which is an important consideration for sufficient network training and testing. It is important to note, however, that while a uniform distribution of data is achieved for the bath temperature range of 955.0 to 985.0°C, temperatures outside this range are extremely rare and therefore difficult to obtain from the smelter knowledge base.

Hence, while there is less representative data for the low and high bath temperature values, all available data for these values is included in the training and test data patterns so that there are some train and test examples for the extremes of the operating range.



(a) (b)

Fig. 5.4.2. Histogram Showing Distribution of the Network Output Variable in the Data Sets. (a) Training, and (b) Test.

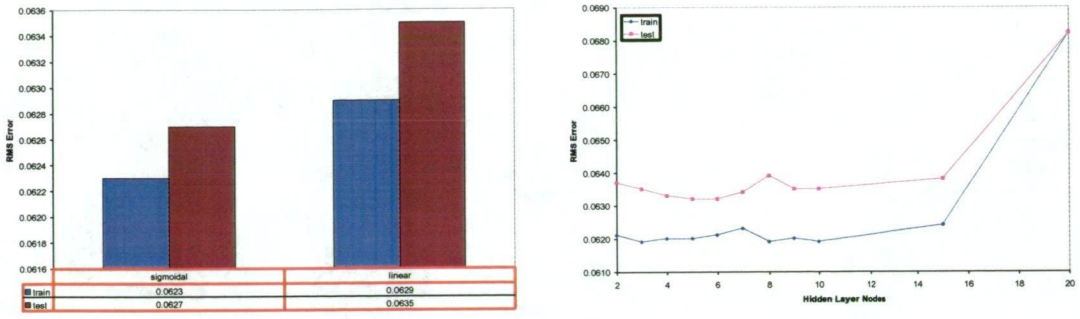
Similar to the previous applications, the MVRA has shown a *t*-critical value of 1.645 to be applicable in this instance. This is determined for a sample size of 1,244, 10 source variables and a confidence level of 95.0%. The significant input variables for this application are highlighted in the results of the MVRA, shown in Table 5.4.1, based on the magnitude of the *t*-statistic, which is required to be higher than 1.645 for a regression coefficient to be significantly different to zero.

TABLE 5.4.1. Multi-Variable Regression Analysis Results for Electrolyte Temperature Prediction Application

Regression Statistics			
r: 0.6432	r^2: 0.4137	*adjusted* r^2: 0.4090	*observations*: 1,244
Parameter	Regression Coefficient	*t*-statistic	Significant
bath temperature (*t*-1)	0.29170	8.45815	yes
bath resistivity	-0.43510	-14.02496	yes
emf	-0.17289	-6.07274	yes
cell age	-0.02599	-2.53354	yes
bath height	-0.07765	-3.41843	yes
high frequency noise	-0.33632	-6.49692	yes
low frequency noise	0.26452	5.42229	yes
cell power	0.04420	1.73155	yes
AlF_3 addition	-0.04402	-2.25444	yes
Na_2CO_3 addition	-0.00780	-0.95338	no

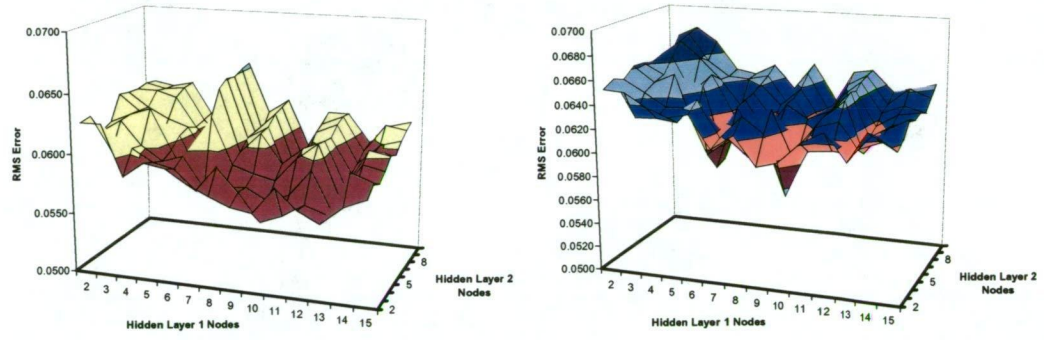
In this instance, the regression statistics for this application have shown an *adjusted* r^2 value of 0.4090. Hence, this infers that approximately 40.9% of the variation of the output variable is explained by the variation of the source variables, while 59.1% of the variation in the output variable is unexplained by this particular regression model. A test of significance of the input variables has shown that, with the exception of Na_2CO_3 addition (t-1), all of the input variables contribute in the regression model to the prediction of electrolyte temperature. Considering the regression coefficients, the percentage contribution of the contributing model variables is determined. It is noted that the most significant parameter in the MVRA model is bath resistivity (25.7%), followed by high frequency noise (19.9%), bath temperature (t-1) (17.2%), low frequency noise (15.6%), emf (10.2%), bath height (4.6%), cell power and AlF_3 addition (2.6%) and cell age (1.5%). In addition, a study of prediction error associated with the regression model has shown that the MVRA yields RMS error values of 0.0654 and 0.0714 for the train and test data sets, respectively. The results of the MVRA model are compared with the neural network modelling results later in this section.

RMS error behaviour for the train and test data sets with changing network architecture is shown in Figures 5.4.3 to 5.4.7, inclusive, and documented in Appendix C, Tables C.41 to C.46, inclusive, for the studied neural network models in this instance. It is shown from a study of error behaviour with changing neural network architecture and algorithm that RMS error is significantly influenced by the number of processing nodes used, the activation function used and the value of the adjustable parameter used in the various activation functions in the applied neural network models. This behaviour is consistent with that observed in the applied neural network models for the applications studied previously. This finding highlights the importance of studying a range of neural network architectures and algorithms in order to determine the minimum error producing model. Notwithstanding, it is useful to note that the minimum RMS error achieved by each of the applied neural networks indicates that a high accuracy model is developed in this instance for electrolyte temperature prediction. The significance of the RMS error associated with neural network modelling for this application in regard to practical implications is discussed later in this section.



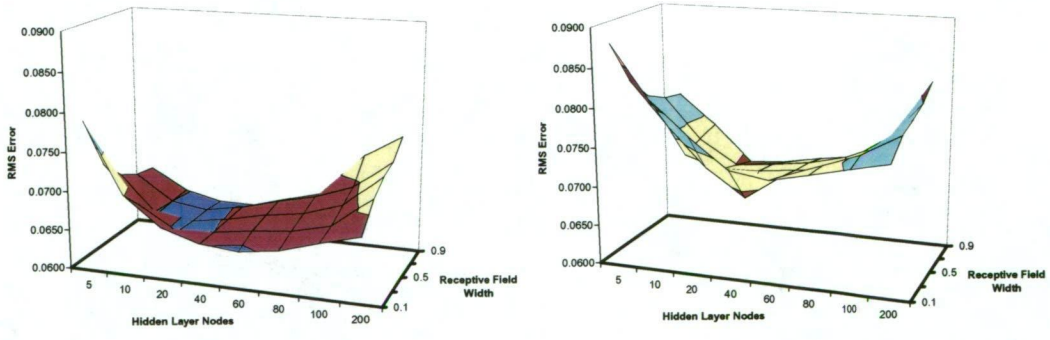
(a) (b)

Fig. 5.4.3. RMS Error Behaviour with Changing Architecture. (a) WH Network, and (b) BP1 Network



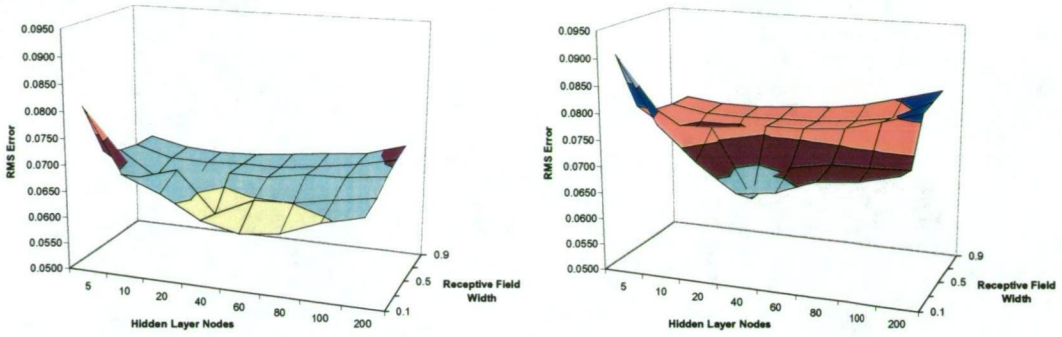
(a) (b)

Fig. 5.4.4. BP2 Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



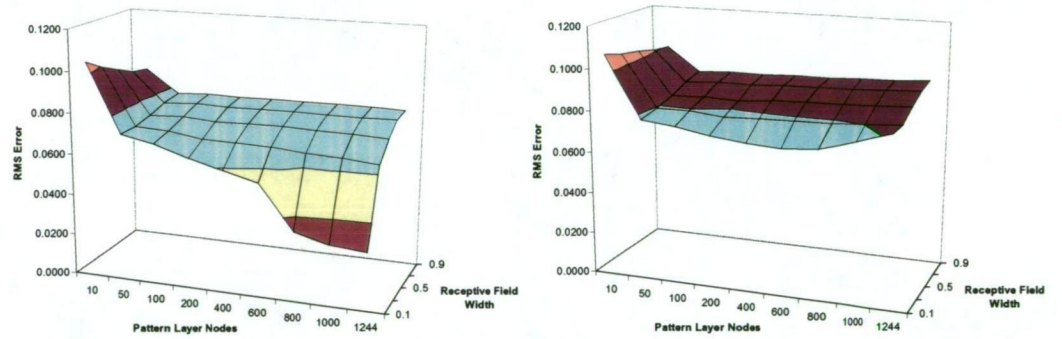
(a) (b)

Fig. 5.4.5. RBF Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.4.6. RBFKOH Network RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test



(a) (b)

Fig. 5.4.7. GRNN RMS Error Behaviour with Changing Architecture. (a) Training, and (b) Test

From the results it is shown that the WH network achieves minimum error when the sigmoidal activation function is used in the output layer of the network. It is shown that a RMS error of 0.0623 and 0.0627 for the train and test data sets, respectively, is achieved using the sigmoidal activation function, which is lower than that of 0.0629 and 0.0635 for the train and test data sets, respectively, obtained using the linear summation function in the output layer. A study of RMS error behaviour with increasing number of iterations, over a range of 1,000 iterations, has shown that the network weights converge after approximately 200 iterations. Considering the BP1 network, using the sigmoidal activation function in the hidden and output layer nodes, it is shown that 5 hidden layer nodes produce the lowest error, yielding RMS error values of 0.0620 and 0.0632 for the train and test data sets, respectively. This is achieved using approximately 650 iterations, which has been found sufficient for weight convergence from a study of RMS error behaviour over a duration of 1,000

iterations. However, the BP2 network has shown a further improvement in RMS error than the WH and BP1 models. It is shown that the BP2 network architecture for achieving minimum RMS error consists of 10 nodes in the first hidden layer and 3 nodes in the second hidden layer. The resulting RMS error using this particular architecture is shown to be 0.0553 and 0.0573 for the train and test data sets, respectively. It was found that approximately 800 iterations were required to allow convergence of the network weights, determined from a study of RMS error behaviour during 1,000 iterations. It is useful to note that the sigmoidal activation function was used in the hidden layer nodes of the BP2 network and also in the output layer nodes. While the activation function used in the hidden layer of the RBF network was the Gaussian function, the output layer activation function used was consistently the linear summation function. Further, it is useful to restate that while the weights connecting the input and hidden layer nodes of this network are fixed, the values of the weights in this instance were set using patterns selected randomly from the training data. It is shown that RMS error is minimum for this model using 30 hidden layer nodes and σ equal to 0.3. Further, it can be seen that the train and test RMS error values produced using this architecture are 0.0629 and 0.0669, respectively, which consequently, is higher than the RMS error achieved using the WH, BP1 and BP2 networks. RMS error behaviour over 1,000 iterations has shown that weight convergence occurs after approximately 250 iterations. Applying the RBFKOH model to this application has shown an improved RMS error from the RBF network. The RBFKOH architecture producing minimum RMS error in this instance is found to be 40 hidden layer nodes with a σ value of 0.3. It is shown that the RMS error associated with this particular architecture is 0.0613 and 0.0645 for the train and test data sets, respectively. In addition, RMS error behaviour with an increasing number of iterations, over a range of 1,000 iterations, has shown RMS error to decrease rapidly during the initial 100 iterations, remaining uniform after approximately 250 iterations are complete. Hence, it is interesting to note from the RBF and RBFKOH prediction error that using the Kohonen clustering technique is beneficial for improving the network accuracy in this instance, compared to selecting network weights randomly from the training data. For the GRNN model it is shown that a minimum RMS error of 0.0549 and 0.0702 for the train and test data sets, respectively, is achieved using 600 pattern layer nodes and σ equal to 0.1. Similar to the RBF and RBFKOH networks, the GRNN model has shown large changes in

RMS error for changing values of σ. This confirms the importance of correctly selecting the value of σ if high accuracy is to be achieved in the neural network models.

A comparison of error associated with each model to determine whether the observed difference is statistically significant is completed using an ANOVA study in the first instance. For ν_1 equal to 6 and ν_2 equal to 1,393, using a significance level of 0.05, a critical test statistic value of 2.1051 is obtained. The results of the ANOVA study for this application are documented in Table 5.4.2. The null and alternative hypotheses for this investigation are that the population means are equal, or at least two population means are unequal, respectively.

TABLE 5.4.2. Test Data Set Error ANOVA Results for Electrolyte Temperature Prediction Application

ANOVA Statistics			
MST: 0.0619	*MSE*: 0.0041	*F*: 15.0594	F_α: 2.1051
Population		Mean	Variance
p_1 - (WH)		-0.0151	0.0040
p_2 - (BP1)		-0.0172	0.0044
p_3 - (BP2)		-0.0077	0.0033
p_4 - (RBF)		0.0201	0.0043
p_5 - (RBFKOH)		0.0284	0.0042
p_6 - (GRNN)		-0.0098	0.0049
p_7 - (MVRA)		-0.0001	0.0040

It is shown that the test statistic, shown to be 15.0594, is greater than the critical value of 2.1051. Hence, the null hypothesis is rejected and it is concluded that at least two of the population means are statistically significantly different. Hence, it is necessary to complete a series of *t*-tests on the paired populations in order to determine which population means are statistically significantly different. The results of the *t*-tests are documented in Table 5.4.3. A *t*-critical value of 2.345 is obtained for a confidence level of 99.0%, or α equal 0.01, and a degree of freedom value of 199. It is shown that the *t*-statistic value for each paired population is greater than the *t*-critical value in each instance. Hence, the null hypothesis is rejected in each instance and it is concluded that all population means are statistically significantly different.

TABLE 5.4.3. Test Data Set Error Statistical *t*-Test Results for Electrolyte Temperature Prediction Application

t_1 to t_7	t_2 to t_7	t_3 to t_7	t_4 to t_7	t_5 to t_7	t_6 to t_7
$t_{12} = 3.662$	$t_{23} = 5.320$	$t_{34} = 13.882$	$t_{45} = 7.569$	$t_{56} = 14.475$	$t_{67} = 4.569$
$t_{13} = 4.378$	$t_{24} = 22.458$	$t_{35} = 17.176$	$t_{46} = 11.073$	$t_{57} = 16.197$	
$t_{14} = 20.543$	$t_{25} = 26.092$	$t_{36} = 3.749$	$t_{47} = 11.906$		
$t_{15} = 23.962$	$t_{26} = 2.918$	$t_{37} = 4.460$			
$t_{16} = 2.503$	$t_{27} = 23.701$				
$t_{17} = 45.090$					

Considering computation time associated with each of the applied neural network models, as documented in Appendix C, Tables C.41 to C.46, a study of computation time for the WH network has shown that the sigmoidal activation function, while producing lower RMS error, gives rise to a computation time of 184.67 seconds for 10 input nodes, which is higher than 116.06 seconds, achieved using the linear summation activation function. For the BP1 network, computation time is shown to increase from that using the WH network, attributed to the inclusion of an additional computational layer in the network. It is shown that for 2 hidden layer nodes, the corresponding computation time is 650.23 seconds, increasing significantly to 2,623.73 seconds for 20 hidden layer nodes. Hence, similar to previous findings, computation time is shown to increase with increasing network complexity. For 2 nodes in either hidden layer of the BP2 network, computation time is shown to be 738.29 seconds, increasing to 5,468.03 seconds for 15 nodes in the first hidden layer and 10 nodes in the second hidden layer. Computation time for the RBF network has shown a significant increase in computation time with increasing number of hidden layer nodes. In particular, for 5 hidden layer nodes and σ equal to 0.3 computation time is shown to be 156.37 seconds, increasing to 2,952.25 seconds for 100 hidden layer nodes. Computation time for the RBFKOH network is shown to be significantly higher than the RBF model, due to the computation time associated with clustering the training patterns using the Kohonen network. It is shown that total computation time for the RBFKOH network with 40 hidden layer nodes and σ equal to 0.3 is 3,195.24 seconds, which is the combination of 1,975.08 seconds for Kohonen clustering, using 1,000 iterations, and 1,220.16 seconds for further RBFKOH training. It is shown that the RBF model with 40 hidden layer nodes and σ equal to 0.3 has an associated computation time of 1,220.17 seconds. For the GRNN architecture achieving minimum RMS error it is shown that computation time is

37.62 seconds for 600 pattern layer nodes and σ equal to 0.1, which is significantly lower than the computation time achieved using the other applied neural networks.

The results of the predictive importance analysis for this application, for each of the applied neural networks, are given in Appendix C, Tables C.47 to C.52, inclusive. In regard to the statistical analysis of the results, it is useful to note that the test data set error was used for comparing population means and further, a *t*-critical value of 2.345 is appropriate in this instance for a significance level of 0.01 and a degree of freedom value of 199. For the WH network it is shown that all of the specified input variables contribute to the prediction of bath temperature, with the exception of emf. It is shown that RMS error is 0.0623 and 0.0626 for the train and test data sets, respectively, when emf is omitted from the WH model. For the BP1 network it is shown that bath resistivity, bath temperature (*t*-1), high and low frequency noise and cell power influence network predictions. However, the remaining variables are not required in the BP1 network for making predictions of electrolyte temperature. Hence, the removal of these non-contributing inputs from the training and test data patterns and re-training the network has resulted in improved RMS error values of 0.0617 and 0.0625 for the train and test data sets, respectively. For the BP2 network the predictive importance analysis has highlighted bath resistivity as the highest contributing input, followed by high frequency noise and bath temperature (*t*-1). Further, it is shown that while the remainder of the inputs all have minor contribution towards the network prediction, they are all shown to be contributing network inputs in this instance. For the RBF neural network it is shown that bath resistivity and bath temperature (*t*-1) are both of similar importance, followed by high frequency noise, then bath height and cell age contributing equally. The remaining variables do not contribute to the network predictions and consequently, are removed from the network model. However, re-training the RBF network with the non-contributing input variables removed yields no improvement in error. An analysis completed for the RBFKOH network has shown that bath temperature (*t*-1) is the most significant contributor of the input variables, followed by bath resistivity and high frequency noise equally. Emf, bath height and cell power are also shown to contribute to the network prediction. Removing the non-contributing input variables from the data patterns and re-training the RBFKOH network produced a slightly improved RMS error of 0.0611 and 0.0640 for the train and test data sets, respectively. For the

GRNN it is shown that all of the input variables contribute to the network prediction, with the exception of AlF_3 and Na_2CO_3 additions. However, re-training the GRNN model with these input variables removed has shown no change in RMS error, producing values of 0.0549 and 0.0702 for the train and test data sets, respectively, which is the same as that achieved with all input variables included in the model.

The results of the casual importance analysis for each of the studied neural network models, as shown in Appendix C, Tables C.53 to C.58, inclusive, have shown similar significance for the contributing input parameters for each model as that obtained using the predictive importance technique. Further, a statistical analysis of the results has shown that each of the contributing input parameters associated with each model exhibit a statistically significant increase in error when varied in the data sets. In each instance it is shown that the *t*-statistic value is higher in magnitude than *t*-critical. Calculating the percentage contribution of the input variables using the results of the predictive and casual importance analyses has produced the results shown in Appendix C, Tables C.59 and C.60, respectively, for each of the applied neural networks. The similarity in ranking of importance of the input variables using the predictive and casual importance techniques is further highlighted in the results of this percentage contribution analysis for each neural network model. A common trend exhibited by the applied neural networks for this application is that bath temperature (*t*-1), bath resistivity and high frequency noise are all shown to be contributing process parameters in each of the neural network models. This useful result indicates that the bath temperature in a reduction cell at time *t*-1 has an influence on the bath temperature of that cell at time *t* and also that bath resistivity and high frequency noise have a high correlation with bath temperature. Similar to the previous applications, the knowledge of these process relationships that has originated from the neural network modelling is critical for understanding process behaviour and improving process control in the aluminium smelting industry.

A comparison of minimum RMS error and computation time achieved with each of the applied network models is shown in Table 5.4.4. Similar to the previous applications, the RMS error shown in the table corresponds to the lowest RMS error achieved with each network by optimising the architecture and removing non-contributing inputs from the model, while the computation time shown is that

associated with the network architecture and number of inputs that produced the lowest RMS error.

TABLE 5.4.4. Comparison of Neural Network Modelling Results for Electrolyte Temperature Prediction Application

	Neural Network Model					
Property	WH	BP1	BP2	RBF	RBFKOH	GRNN
RMS error - train	0.0623	0.0617	0.0553	0.0616	0.0611	0.0549
RMS error - test	0.0626	0.0625	0.0573	0.0665	0.0640	0.0702
comp. time (s)	166.9	637.2	2,480.2	897.7	3,117.8	34.9

It is interesting to note in this instance that the neural network model achieving the lowest RMS error, shown to be the BP2 network, is also the only model that uses all of the 10 specified network inputs as contributing model variables. The BP1 network has shown a slightly higher RMS error than the BP2 network, however, it only uses 5 of the specified network inputs, while the WH network has 9 of the specified input parameters as contributing model variables but has a higher associated RMS error than the BP1 network. It is interesting to observe that, similar to the previous applications, the RBF network has a lower RMS error when the network weights are clustered using the Kohonen network, compared to selecting randomly from the training data, and further that the ranking of importance of the input variables is different between the RBF and RBFKOH networks. The GRNN model has shown the highest test RMS error of the applied neural networks for this application and shown that 8 of the input variables are used in the GRNN predictions of electrolyte temperature. It is interesting to note that the neural network modelling completed in this instance has again shown lower RMS error than the regression model developed for this application. It is shown that while a RMS error of 0.0654 and 0.0714 is obtained for the train and test data sets, respectively, using the MVRA, a lower train and test RMS error is achieved with each of the applied neural network models. In particular, the BP2 network, which has shown the lowest RMS error of the applied neural network models in this instance, achieved a significantly higher accuracy than the MVRA technique. Further, it is shown that the regression model required 9 of the specified input parameters for predictions of electrolyte temperature, with Na_2CO_3 addition being shown as the only non-contributing model variable.

In order to study the practical implications of the neural network modelling completed in this instance it is useful to compare actual and predicted values of electrolyte temperature. A useful technique for this comparison is a dispersion plot of actual and predicted electrolyte temperature values, as shown in Figure 5.4.8.

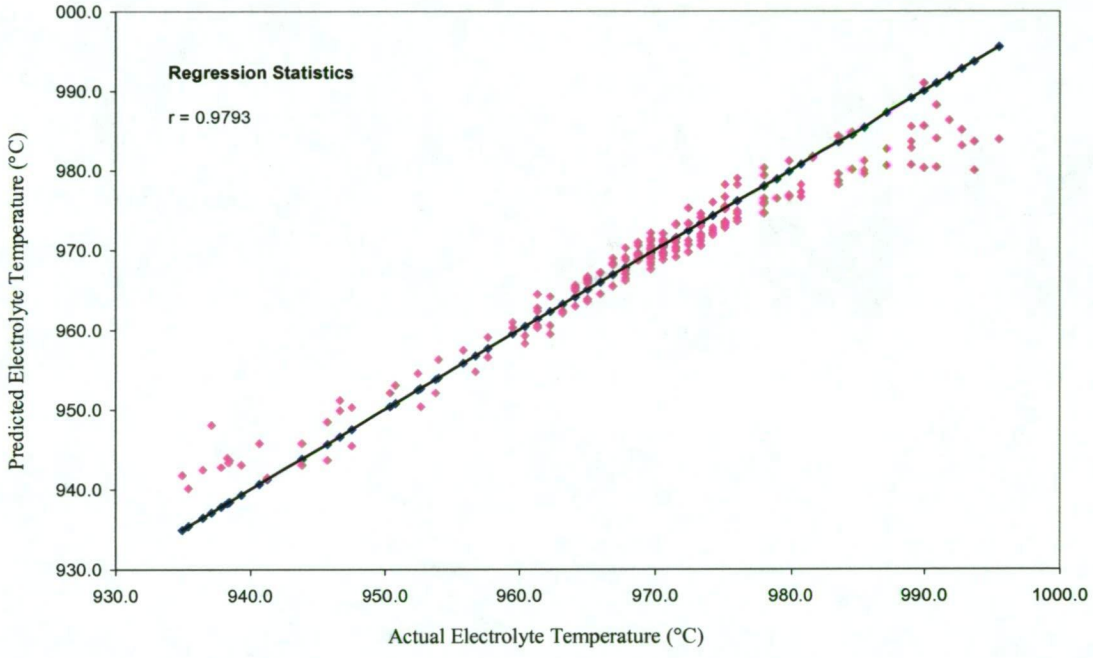


Fig. 5.4.8. Dispersion Plot for BP2 Neural Network for Electrolyte Temperature Prediction Application

It is important to note that electrolyte temperature predictions are completed using the BP2 model in this instance as it has shown the lowest RMS error of the applied models for this particular application. The dispersion plot shows the deviation of predicted electrolyte temperature values from the target test data set output values. The trend-line added to the dispersion plot highlights the target output values for the corresponding actual electrolyte temperature and therefore has a correlation coefficient of 1.0; it is linear and has the regression equation $y = x$. On the other hand, the predicted values are shown to have some dispersion from the target trend-line, indicating error in the neural network predictions. It is can be seen that while the predicted and actual values are similar over the temperature range 955.0 to 985.0°C, the prediction error is higher outside this range. This behaviour indicates a correlation between prediction error and the number of training data patterns. It has been shown that while the temperature range of 955.0 to 985.0°C is well represented

in the training data, temperatures outside this range are rare and therefore less data is included for the extremes of the operating range. Consequently, the error associated with temperature predictions at the extremes of the operating range have a higher associated error due to the lack of sufficient training for these minority instances. Nevertheless, the correlation coefficient, r, of 0.9793 noted on the dispersion plot confirms a strong correlation between actual and predicted electrolyte temperature values. Specifically, the closer the value of the correlation coefficient is to 1.0 the higher the accuracy of the neural network. In this instance, the high value of the correlation coefficient indicates that the developed BP2 model is relatively accurate for electrolyte temperature predictions.

An approximately normal distribution of prediction error is shown to be associated with the neural network in this instance, as shown in Figure 5.4.9.

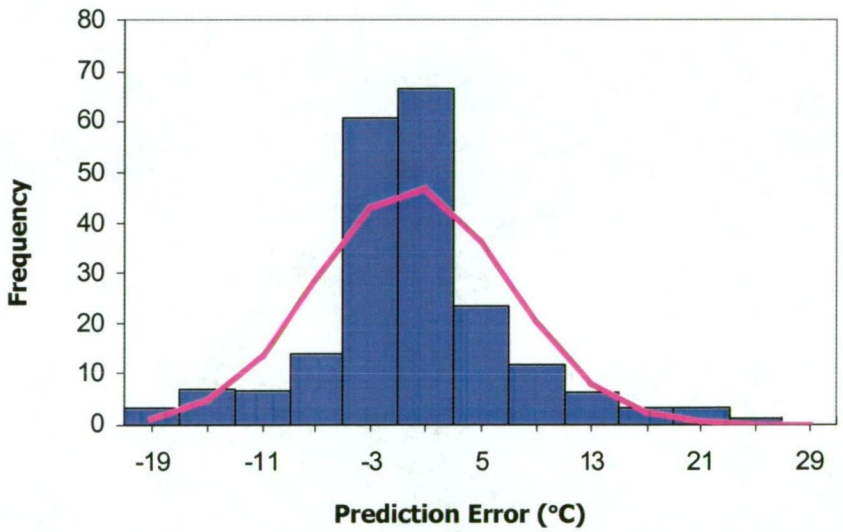


Fig. 5.4.9. Histogram Showing Prediction Error Distribution for Electrolyte Temperature Prediction Application

It can be seen that a significant number of predictions of electrolyte temperature have approximately zero error, while there is an equal distribution of predictions that are higher and lower than the actual values. Moreover, the normal curve approximation of the histogram is shown to represent a normal distribution symmetrical about the mean, confirmed by a skewness value of -0.01°C, as shown in Table 5.4.5 along with further informative histogram statistics. While it is shown that the maximum

prediction error has a magnitude of 13.70°C, it is noted that the mean prediction error, or 50th percentile, is -0.03°C and that approximately 50.0% of the prediction error is lower than 0.0°C, ie. the neural network is not biased towards higher- or lower-than-actual predictions.

TABLE 5.4.5. Prediction Error Histogram Statistics for Electrolyte Temperature Prediction Application

Descriptive Statistics	Statistical Value
data size (test)	200
minimum value (°C)	-13.70
maximum value (°C)	11.04
range (°C)	24.74
mean value, or 50th percentile (°C)	-0.03
standard deviation (°C)	7.02
skewness (°C)	-0.01
proportion of error below 0.0°C	0.50

The average prediction error associated with the nominated ranges for electrolyte temperature, as shown in Table 5.4.6, confirms the existence of a correlation between prediction error and the number of training data patterns for a particular range of data. The percentage of bath temperature measurements in the nominated ranges are also noted in the table. All bath temperature measurements over a six-month period at the smelter were analysed to determine the frequency of bath measurements in each of the nominated bath temperature ranges and hence, calculate the percentage of values in each range. It is shown that average prediction error is low for those electrolyte temperature ranges that are well represented in the training data set. In particular, it has been shown that the temperature range of 955.0 to 985.0°C is well represented in the training data patterns, which consequently has shown a low prediction error. On the other hand, prediction error is comparatively high for electrolyte temperatures lower than 955.0°C and greater than 985.0°C. Nevertheless, for approximately 90.0% of all electrolyte temperature measurements electrolyte temperature is found to be in the range of 955.0 to 985.0°C, for which the neural network modelling has shown a high degree of accuracy. It is useful to note that, similar to the electrolyte additive prediction application, average absolute error is documented in the table. For practical implementation, it is important to study the accuracy of the neural network by investigating the magnitude of the difference

between actual and predicted values, rather than an average error value that incorporates positive and negative error, as this average is typically zero for a non-biased neural network.

TABLE 5.4.6. Prediction Error Associated with Specified Data Ranges for Electrolyte Temperature Prediction Application

Bath Temperature Range (°C)	Average Prediction Error in this Range (°C)	Percentage of Temp. in this Range (%)
935.1 to 945.0	10.93	0.4
945.1 to 955.0	4.45	9.2
955.1 to 965.0	2.70	46.1
965.1 to 975.0	2.10	31.5
975.1 to 985.0	4.90	11.7
985.1 to 995.0	11.90	1.1

In order to compare the error associated with the neural network modelling completed for this application with the existing practice for determining electrolyte temperature in the reduction cell it is useful to highlight a recent study that was completed at CABBL. It has been shown in the previous chapter that a recent investigation completed at CABBL has shown manual electrolyte temperature measurements to have an associated error of ±4.0°C, attributed equally to equipment error and operator error. Hence, the neural network modelling results have shown comparable error to the existing technique for determining electrolyte temperature. Therefore, there is confidence in implementing the neural network to further quantify the accuracy of the process model for this application. The results of this implementation are documented in a later chapter.

The range of neural network modelling results obtained for this industrial application has further confirmed the requirement for a neural network selection methodology that is capable of decision making based on economic consideration. It is necessary to select a neural network for this application that is most economically beneficial for the task, considering the associated model error, the process variables required for the model and the computation time required to develop the process model.

5.5 CONCLUDING REMARKS

It is important to note that the cause and effect are fixed in any physical process. Further, there may be multiple causes for the same effect. If several process

parameters are correlated with some particular performance feature, then it is likely that there will be some process parameters that have more influence than others on that performance feature. Although the physics and chemistry of the process may be well understood, the degree of influence of each of such possible process parameters on the performance need not be well known. Depending upon the characteristics and behaviour of the developed neural network models, the range and extent of input parameter influence on the network decision will change, as highlighted in the studied industrial applications in this instance. It is shown for each application that the influence of the specified input parameters is dependent on the applied neural network model. While this does not change the physics and chemistry of the process, it gives a better understanding of which model and corresponding inputs to select as mere mathematical tools that best represent the process.

The high accuracy achieved with the neural network modelling completed for each of the studied industrial applications documented in this chapter gives confidence in further applying the developed models on-line at CABBL to achieve process improvements. While the rationale for applying neural networks for the studied applications has been provided in a previous chapter it is necessary to implement the developed models to achieve these noted benefits. While the work documented in this chapter involves the development of appropriate models in each instance, it has been discussed that is necessary to select the most appropriate model for each application based on economic consideration. For each of the industrial applications considered in this chapter it is shown as a result of neural network modelling that the behaviour of the applied neural network models is significantly different in each instance. It is shown that RMS error, ranking of importance of the input variables and computation time for each application are significantly different for each of the applied neural networks. Moreover, while a particular neural network model achieves lower RMS error for a particular application it does not necessarily achieve the lowest RMS error for all applications. While the GRNN achieved highest accuracy for the electrolyte additive prediction application, the BP1 network achieved the lowest error for the cell failure prediction application and the BP2 network for the electrolyte temperature prediction application. Moreover, it is interesting to note that the GRNN obtained the highest error for the cell failure prediction application and the electrolyte temperature prediction application. Hence, it is shown that a particular

neural network is not capable of achieving the lowest prediction error for all applications, but rather, the most suitable neural network for an application is dependent on the characteristics and particularities of the application.

It is important to note in these concluding remarks that a significant result to emanate from the neural network modelling completed for each of the studied industrial applications is an improved understanding of the aluminium smelting process. This has been achieved by considering the ranking of importance of the input variables specified in each instance for the industrial applications studied as a part of this work. The knowledge and understanding of these previously unknown relationships is fundamental for the development and improvement of process control techniques at CABBL. While neural networks are often perceived as a 'black-box' modelling strategy, the adoption of techniques, such as the predictive and casual importance strategies introduced and used during this investigation, provide valuable process information. Further, it is also important to note that the results achieved in this instance provide confidence in applying neural network modelling to further applications at CABBL. The generic nature of the neural network algorithms and the dynamic behaviour of the aluminium smelting process make neural network modelling well suited to this industry. The investigation completed for the particular industrial applications studied as a part of this work has established a basis for completing further neural network modelling at CABBL and its associated companies and has significantly contributed towards introducing and establishing these useful computational tools as a process control methodology.

It has been noted throughout this chapter that there is a requirement to develop a neural network selection methodology that allows a neural network to be selected for a particular application based on economic consideration. In particular, it is necessary to develop a technique that will allow a neural network to be selected that will maximise the economic benefit of the modelling methodology. While it has been noted in a previous chapter it is useful to restate here that the selection methodology must consider the following:

i). network model accuracy and cost associated with accuracy of network prediction

ii). input variables required for network model and cost associated with obtaining data for required variables, and

iii). computation time required by the network model to converge to a solution and the cost associated with this time

The following chapter addresses each of these noted assessment criteria for neural network model selection and further, discusses the requirements of an optimisation technique appropriate for decision making based on economic consideration.

CHAPTER SIX

Criteria for Model Selection and Assessment of Optimisation Techniques

6.1 INTRODUCTION TO ASSESSMENT CRITERIA FOR NEURAL NETWORK SELECTION

There is a need to develop a neural network selection methodology based on economic considerations, such as; cost associated with model accuracy, cost associated with obtaining data for required model variables and associated cost of model computation time. In particular, it is now useful to introduce some specific assessment criteria that are required for selecting a suitable neural network for a specific application. It is important to note that the assessment criteria listed here are considered necessary and satisfactory to make a decision of which neural network is most economically viable for each of the studied industrial applications.

1. RMS error - For this particular assessment criterion the applied models are sorted based on the RMS error achieved using the optimum model architecture and algorithm and further, when all non-contributing input variables are removed from the neural network models. An example is provided to illustrate this assessment criterion. Referring to Figure 6.1.1, it can be seen that the minimum test RMS error achieved using activation function A is 0.0970, achieved using 8 hidden layer nodes. However, minimum test RMS error decreases to 0.0758 using activation function B and the same number of hidden layer nodes. Hence, activation function B and 8 hidden layer nodes are used to complete an importance analysis as this particular architecture and algorithm yield minimum error. As a result of this analysis it is found that some of the input parameters are non-contributing and are consequently removed from the model, resulting in an improved test RMS error of 0.0703, for example. Hence, this particular RMS error is the minimum RMS error that can be achieved by the model and represents the RMS error used for this assessment

criterion. Hence, the applied models are ranked in order of increasing RMS error; the neural network ranked first is the model that achieves minimum RMS error subsequent to selection of the minimum error producing architecture, algorithm and contributing number of input parameters.

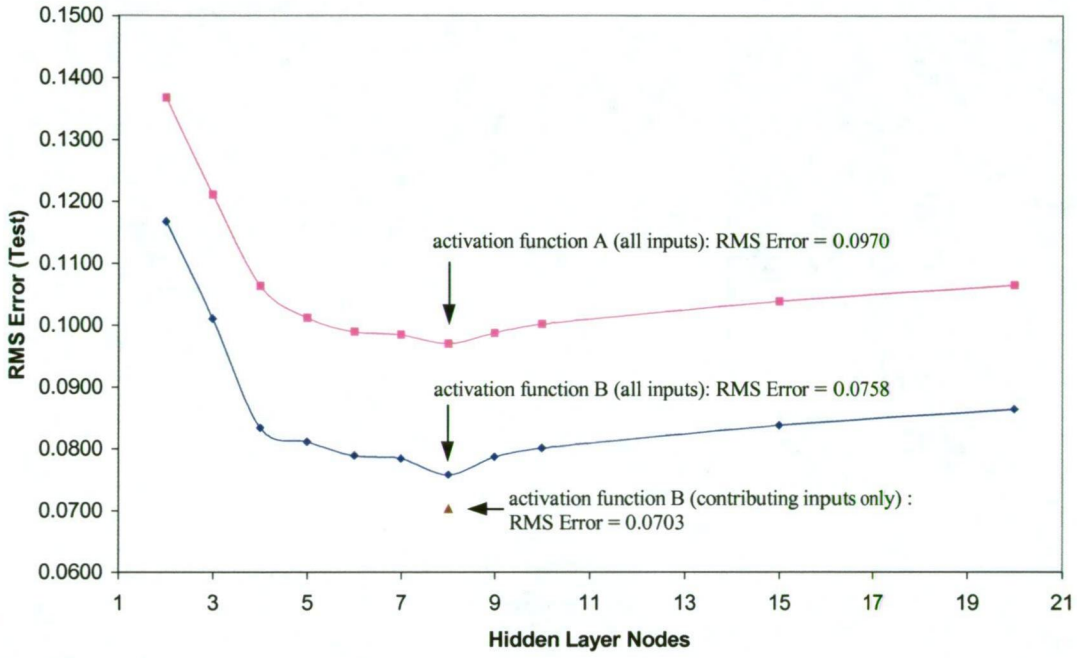


Fig. 6.1.1. Illustration of Minimum Test RMS Error Used as Assessment Criterion for Model Selection

2. Number of input variables required - It has been shown for each of the studied applications that the number of input variables required by the applied neural networks to develop a process model is different in each instance. It is especially relevant to note that it is economically beneficial to minimise the number of input parameters used in a neural network model. In particular, it has been shown that the complexity of a neural network model reduces with reducing number of inputs, consequently, reducing the required computation time for neural network convergence. Further, the total cost associated with process parameter measurement is lower if less input variables are required to predict the performance feature, as there are a lower number of process parameters to measure. Hence, it is economically beneficial to use the minimum number of parameters required to specify the process behaviour. Specifically, those covariates that are shown to be not significant are removed from the full model in order to reduce the number of input variables.

Consequently, a more parsimonious model is developed without compromising the accuracy of the neural network. Hence, the objective of the selection criteria in this instance is to establish the smallest subset of the covariates that yields minimum prediction error. Further, it is important to note that the number of input variables selected for each model in each instance is equal to the number of input parameters required to achieve the minimum test RMS error. For example, referring to Figure 6.1.1, the dimension of the subset of covariates is equal to the number of input variables required to achieve a test RMS error of 0.0703; it consists of contributing input parameters only. Hence, the applied models are sorted for each application based on the number of input parameters required by each model, in order of descending suitability, with most suitable representing the model requiring the least number of input parameters.

3. Computation time - This assessment criterion considers the computation time required by each of the neural networks to develop a process model. That is, the time required by each model to complete a sufficient number of iterations such that a converged weight matrix is established, where further iterations would not decrease RMS error, as demonstrated in Figure 6.1.2.

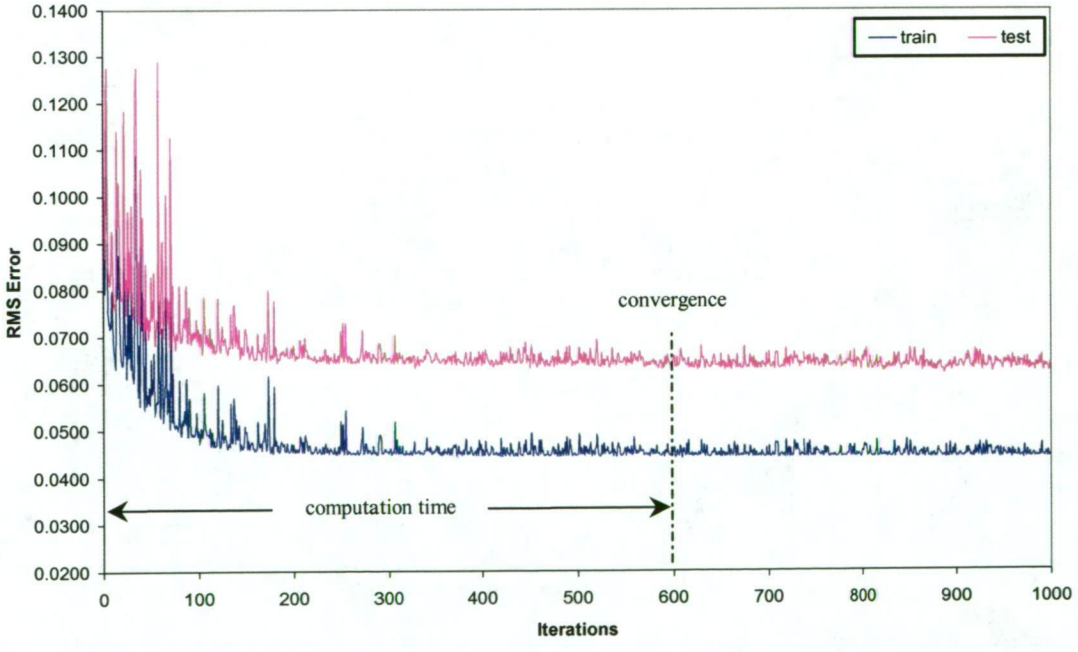


Fig. 6.1.2. Plot of RMS Error Behaviour with Increasing Iterations to Illustrate Associated Computation Time

It is shown that RMS error remains unchanged after approximately 600 iterations. Hence, the computation time used for this assessment criterion is the time required to achieve model convergence. For this assessment criterion, the neural networks are ranked in order of ascending computation time, with the most suitable model in each instance being that with the lowest associated computation time.

While the applied neural networks are ranked in the following work according to these specified assessment criteria, it is also interesting to consider the MVRA modelling results as a part of this investigation, to provide a comparison with the neural network models.

6.2 EVALUATION OF NEURAL NETWORK MODEL PERFORMANCE BASED ON ASSESSMENT CRITERIA

For each of the studied industrial applications, the applied neural network and multi-variable regression analysis models are ranked in order of descending suitability, based on the specified assessment criteria. It is important to note that the ranking in this instance is completed using the modelling results documented in the previous chapter. While a model selection methodology is not discussed here, although it is introduced in future work, this analysis is necessary to highlight the complexities associated with selecting a particular neural network for a specific application, when economic benefit is the driving force for model selection. Moreover, the model ranking completed here, using the given assessment criteria, is critical for developing a strategy for neural network selection.

6.2.1. Model Ranking Based on RMS Error

The model achieving minimum RMS error is ranked as most suitable for the specific application, while the model with the highest associated RMS error is the least suitable, with the remaining models ranked between these extremes. The model ranking from lowest RMS error through to highest is documented in Table 6.2.1 for each of the studied industrial applications. It can be seen that while the GRNN model achieved minimum RMS error for the electrolyte additive prediction application, the BP1 network has shown the lowest error for the cell failure prediction application and the BP2 model for the electrolyte temperature prediction application. The results documented here highlight the importance of applying and studying a range of

different neural networks for each specific application, not selecting a particular model based solely on its success with previous applications. It is shown that no single model achieves the minimum RMS error for all applications.

TABLE 6.2.1. Ranking of Models Applied to Industrial Applications Based on RMS Error

	Electrolyte additive prediction application		Cell failure prediction application		Electrolyte temperature prediction application	
Model Ranking	Model	RMS Error	Model	RMS Error	Model	RMS Error
1st	GRNN	0.0671	BP1	0.0451	BP2	0.0573
2nd	RBFKOH	0.0715	BP2	0.0580	BP1	0.0625
3rd	BP2	0.0719	WH	0.0761	WH	0.0626
4th	RBF	0.0727	RBFKOH	0.1074	RBFKOH	0.0640
5th	BP1	0.0774	RBF	0.1279	RBF	0.0665
6th	WH	0.1121	GRNN	0.1598	GRNN	0.0702
7th	MVRA	0.1129	MVRA	0.3264	MVRA	0.0714

6.2.2 Model Ranking Based on Number of Input Variables Required

The model ranking from lowest number of input variables required through to highest number of input variables is documented in Table 6.2.2 for each of the studied industrial applications. While some input variables are shown to have a low percentage contribution in some instances, these input parameters are nevertheless included as contributing network inputs in the following ranking for this assessment criterion. It is shown that a low percentage contribution for a particular input parameter corresponds to a low change in RMS error if that particular input is omitted from the model. However, this change, while only small, is considered significant in this instance as it does indeed decrease model accuracy. Moreover, the removal of many parameters with a low percentage contribution has a cumulative effect on RMS error, decreasing model accuracy substantially if many low percentage contribution parameters are omitted from the model.

Similar to the model ranking based on RMS error, it can be seen that the model ranking for this assessment criterion is significantly different for each of the studied industrial applications. While it is shown that the GRNN model required the lowest number of parameters for the electrolyte additive prediction application, it is shown that the MVRA model required the minimum number of input variables for the cell

failure prediction application and the BP1 network for the electrolyte temperature prediction application.

TABLE 6.2.2. Ranking of Models Applied to Industrial Applications Based on Number of Input Variables Required

	Electrolyte additive prediction application		Cell failure prediction application		Electrolyte temperature prediction application	
Model Ranking	Model	Number of Inputs	Model	Number of Inputs	Model	Number of Inputs
1st	GRNN	5	MVRA	8	BP1	5
2nd	MVRA	6	GRNN	12	RBF	5
3rd	WH	7	RBFKOH	14	RBFKOH	6
4th	RBFKOH	8	RBF	18	GRNN	8
5th	BP1	9	WH	20	WH	9
6th	RBF	10	BP2	22	MVRA	9
7th	BP2	12	BP1	23	BP2	10

6.2.3 Model Ranking Based on Computation Time

It has been noted that the computation time used to rank the applied models in this instance is the minimum time required by each model to achieve the minimum RMS error documented in Table 6.2.1. Hence, the computation time noted in each instance is the minimum achievable by each of the applied models to obtain the documented minimum RMS error. The ranking of the applied models for each of the studied industrial applications is shown in Table 6.2.3 in order of increasing computation time.

TABLE 6.2.3. Ranking of Models Applied to Industrial Applications Based on Computation Time

	Electrolyte additive prediction application		Cell failure prediction application		Electrolyte temperature prediction application	
Model Ranking	Model	Time (s)	Model	Time (s)	Model	Time (s)
1st	MVRA	4.5	MVRA	5.0	MVRA	4.2
2nd	GRNN	24.6	GRNN	24.8	GRNN	34.9
3rd	WH	281.0	WH	271.3	WH	166.9
4th	BP1	1,010.3	RBF	1,793.3	BP1	637.2
5th	RBF	1,319.4	BP2	2,871.1	RBF	897.7
6th	RBFKOH	2,711.1	BP1	4,438.2	BP2	2,480.2
7th	BP2	3,089.1	RBFKOH	4,547.4	RBFKOH	3,117.8

Ranking of the applied models based on computation time is consistent with model complexity and the number of iterations required by each of the models to achieve convergence. Hence, the MVRA model, which is shown to have low complexity and requires no computational training iterations has a low computation time. Similarly, the GRNN model, although of higher complexity than the MVRA model, also requires no computational training iterations and therefore has a low associated computation time. However, the higher complexity models such as BP2, RBF and RBFKOH, requiring a relatively large number of computational training iterations, have shown high computation times and are therefore ranked after the MVRA, GRNN and WH models in terms of suitability for this particular assessment criterion.

6.3 COMPLEXITY OF MODEL SELECTION BASED ON ASSESSMENT CRITERIA

While the model ranking has been completed for each of the specified assessment criteria, it is useful to tabulate the results of this ranking for each of the studied industrial applications, as shown in Table 6.3.1. The objective of this is to highlight for each application that a different model ranking is achieved for each of the assessment criteria used. Hence, for each industrial application there is no particular model that is an obvious choice as most suitable.

TABLE 6.3.1. Ranking of Models for Studied Industrial Applications Based on Specified Assessment Criteria

Model Ranking	RMS Error	Number of Input Variables	Computation Time
	Electrolyte Additive Prediction Application		
1st	GRNN	GRNN	MVRA
2nd	RBFKOH	MVRA	GRNN
3rd	BP2	WH	WH
4th	RBF	RBFKOH	BP1
5th	BP1	BP1	RBF
6th	WH	RBF	RBFKOH
7th	MVRA	BP2	BP2
	Cell Failure Prediction Application		
1st	BP1	MVRA	MVRA
2nd	BP2	GRNN	GRNN
3rd	WH	RBFKOH	WH
4th	RBFKOH	RBF	RBF
5th	RBF	WH	BP2
6th	GRNN	BP2	BP1

7th	MVRA	BP1	RBFKOH
	Electrolyte Temperature Prediction Application		
1st	BP2	BP1	MVRA
2nd	BP1	RBF	GRNN
3rd	WH	RBFKOH	WH
4th	RBFKOH	GRNN	BP1
5th	RBF	WH	RBF
6th	GRNN	MVRA	BP2
7th	MVRA	BP2	RBFKOH

It can be seen from the analysis completed that the ranking of the developed neural network models for each of the studied industrial applications is different in each instance. Further, for each application there is no single model that is ranked first for each assessment criterion. Moreover, the decision of which neural network model to select for a specific application is complex as the assessment criteria are not fundamentally of equal significance in each instance. Rather, the influence of each assessment criterion is highly dependent on the application considered. For instance, RMS error maybe of high significance for a particular application where high prediction accuracy is critical, but of low significance for an application where high prediction accuracy is not essential. Nevertheless, economic benefit is the major consideration for model selection. It is necessary for each application to complete an analysis of each model to determine the optimum combination of the selection criteria that yields minimum cost per prediction. While it has been shown that error increases with the removal of contributing input parameters from the model, it is important to note that the cost associated with an increase in error may be lower than the cost associated with measurement of the contributing process parameter. In particular, it may be economically beneficial to remove an input parameter from a model, even though it has been shown to contribute to minimising prediction error, if the cost of measuring the input parameter is greater than the cost associated with the increase in error that results from the removal of the input parameter from the model. Hence, a decision model is required to analyse the decision problem using systematic identification and evaluation of all the available alternatives. The decision of the optimum model for each application is then reached by selecting the alternative that provides maximum economic benefit. Hence, it becomes necessary to complete the decision of neural network selection using appropriate optimisation techniques, as introduced in the following section.

6.4 STRATEGY FOR DECISION MAKING FOR APPROPRIATE NEURAL NETWORK SELECTION

Typically in industry, in order to minimise production costs it is important to achieve maximum economic benefit from the resources available. Hence, it is necessary to have some procedure for making a decision of which particular resource will provide maximum economic benefit. For this purpose, the scientific analysis provided by operations research techniques is one particular possible optimisation methodology. Operations research is the study of the application of mathematical techniques to select from various alternatives that decision or decisions that will maximise some quantitatively measured goal [198]. Further, a numerical measure of effectiveness is necessary for operations research techniques, during which the optimisation problem is expressed in terms of mathematical relationships and subsequently solved by means of computational methods [199]. The strength and versatility of operations research stems from its diagnostic power through observations and modelling as well as from its perspective power through analysis and synthesis [200]. In particular, the contribution from operations research techniques is primarily attributed to the following characteristics [201]:

i). Structuring the situation into a mathematical model, abstracting the essential elements so that a solution relevant to the decision maker's objectives can be sought.

ii). Exploring the structure of such solutions and developing systematic procedures for obtaining them.

iii). Developing a solution, including the mathematical theory, if necessary, that yields an optimal value of the system measure of desirability, or possibly, comparing alternative courses of action by evaluating their measure of desirability.

Hence, the inherent systems and methodologies associated with operations research indicate that this particular decision making technique may be appropriate for neural network model selection. Therefore, it is necessary to investigate the characteristics and particularities of operations research techniques, as introduced in the following section.

6.5 EVALUATION OF OPERATIONS RESEARCH TECHNIQUES

Operations research simply defined means 'scientific approach to decision making' [202]. It seeks the determination of the best course of action of a decision problem under the restriction of limited resources [203]. Operations research is associated almost exclusively with the use of mathematical techniques to model and analyse optimisation problems [204]. An optimisation problem involves choosing from many alternatives the one that yields a maximum or minimum value of some numerically measurable criterion of performance [205]. The problem of translating an optimisation problem into mathematical form is called 'formulating' the problem. Having formulated, there are techniques that must be applied to solve the problem. The methods and algorithms for doing so are broad and range in difficulty from simple to complex. Of all the techniques of operations research, linear programming [202, 203, 206] is by far the best known and most widely practiced, attributed mostly to the solution methods having the virtues of being easily understood and highly efficient [205]. The term linear programming defines a particular class of programming problems that meet the following conditions [207]:

i). The decision variables involved in the problem are non-negative, ie. they are either positive or zero. Further, the decision variables are divisible into fractional parts as well as whole units.

ii). The criterion for selecting the 'best' value of the decision variables can be described by a linear function of these variables, that is, a mathematical function involving only the first powers of the variables with no cross products. Hence, the association between decision variables is additive.

iii). The operating rules governing the process can be expressed as a set of linear equations or linear inequalities.

Hence, the term linear programming is used to define a methodology for modelling problems so that they have a special structure, while it also denotes the strategy for solving problems with such a structure [208]. It is essentially a geometric or algebraic procedure whereby the optimum allocation of a parameter, where there are two or more alternatives, is determined in regard to predefined objectives and specified constraints or conditions [209]. Hence, linear programming basically attempts to optimise an objective function subject to a set of constraints [210]. Therefore,

formulation of the optimisation problem into linear programming form requires defining the variables involved and establishing relationships between them to determine the objective function and associated constraints. The objective function expresses the value system or goal, while the constraints are limitations of the environment or capabilities that limit the degree to which the objective function can be pursued [211]. The problem objective function and constraints must be represented as mathematical expressions in order to be solved using linear programming techniques [212]. Hence, the objective function and associated constraints are written in the following standard form [206]:

$$\begin{aligned} \text{Maximise,} \quad & z = c_1x_1 + c_2x_2 + \ldots + c_nx_n && (\textit{objective function}) \\ \text{subject to,} \quad & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 && (\textit{constraints}) \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2 \\ & \cdot \\ & \cdot \\ & a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m \\ \text{and,} \quad & x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0 \end{aligned} \tag{6.5.1}$$

The variables x_1 to x_n are called the 'decision variables' of the linear programming model, while the values of c_1 to c_n and a_{11} to a_{mn} are the coefficients of the decision variables. Further, the values of the variables x_1 to x_n are said to constitute a feasible solution if they satisfy all the specified constraints [204]. While a feasible solution is one that satisfies all of the constraints, an optimal solution is the best solution from among all the feasible solutions [213]. Once the linear programming problem has been appropriately formulated it can be solved using either a graphical technique [204, 208, 213, 214] if the problem involves only two decision variables or the simplex method [211, 214, 215] for a higher number of decision variables. While the graphical technique is useful for two-variable linear programming problems, the majority of problems encountered have many variables, hence, the simplex method is typically required [199]. The simplex algorithm for solving linear programming problems was developed by George Dantzig [216] in 1947. It is a linear programming technique that uses a successive improvement method and formal procedure to ensure that a better solution will be obtained after each iteration of the simplex algorithm and an optimal solution will be obtained in a finite number of

iterations [211]. The simplex algorithm starts at a basic feasible solution point and tests this point for optimality, which is a test to determine if the current solution is an optimal solution. If the current solution is not optimal, the algorithm indicates a feasible move for improvement, whereby all moves are to a basic feasible solution, since an optimum only exists at such points [214]. However, an assumption of the simplex method is that the decision variables are divisible, that is, the coefficient of each decision variable can be expressed as a fraction, or real number. In spite of this, it is important to note and will be shown in the following chapter, that some of the decision variables used in the linear programming models developed for each industrial application have physical significance only if they have integer, or whole number, values. However, it is quite possible that the simplex method will assign non-integer values to these decision variables when specifying an optimal solution. Nevertheless, in order to deal with this special class of linear programming problems, where integer values are required for some decision variables, a particular technique known as integer programming has been developed. Some problems require all decision variables to be assigned integer values, called 'all-integer programming', while the majority of problems require only some decision variables to be integer, the remaining non-integer, called 'mixed-integer programming' [217]. However, integer programming solution procedures, such as the cutting plane technique [218, 219] and the branch-and-bound method [220,221], are very inefficient and therefore undesirable to use for problems involving many decision variables, such as the problem of model selection for the studied industrial applications.

6.6 RATIONALE FOR DEVELOPMENT OF AN OPTIMISATION TECHNIQUE FOR MODEL SELECTION

While the techniques associated with operations research, such as identification of appropriate decision variables and development of an objective function, are necessary concepts that are useful for problem formulation for neural network model selection, it is highlighted in the following discussion that available operations research techniques do not provide an appropriate optimisation methodology in this instance. Moreover, the following discussion provides a rationale for development of an alternative optimisation technique for model selection.

The requirement that all decision variables must be able to be expressed as fractions in the optimal solution has been identified as a limitation of the simplex method. Due to the high number of decision variables associated with each of the industrial applications that are required to be assigned an integer value, or more specifically a value of either 0 or 1, if the solution specified by the simplex method is unable to be rounded appropriately to obtain an optimal solution then it will be necessary to apply integer programming techniques to obtain the desired optimal solution. However, such techniques are inappropriate in this instance due to the high number of decision variables associated with the studied applications, making the available integer programming techniques highly inefficient. Further, it has been shown that constraint specification is a critical requirement for linear programming model development. The constraints associated with each of the industrial applications are determined by considering the limitations of the process in each instance and subsequently, representing these limitations in mathematical form as linear equations or linear inequalities. However, while the developed objective function for each application is common for all models it is necessary to develop a unique set of constraints for each model. An explanation of this requirement is aided by considering the necessary constraints. In particular, certain constraints are required for each model to impose the necessary limitations on the objective function developed for each application. A discussion of the required constraints is given in the following:

i). A constraint is required to place a bound on the minimum RMS error each model can achieve. Without this constraint the linear programming model may suggest zero error is optimum for each model, however, it is not practicably possible to achieve this result. Hence, the constraint specifies that the decision variable representing RMS error must be greater-than-or-equal-to the minimum RMS error value achieved by each model. Therefore, while RMS error greater than the minimum achieved may be determined by the linear programming model as optimum, it cannot be lower than the minimum achieved using all contributing input parameters.

ii). It is necessary to develop a constraint that specifies that non-contributing input parameters, as determined for each model, are not to be considered in the optimal solution. Non-contributing input parameters provide no economic benefit as they have an associated measurement cost, increase computation

time and do not reduce error. Hence, they are redundant parameters in the model and must be excluded.

iii). The increase in RMS error that results from the removal of contributing input parameters from a model, as given by the predictive importance analysis, is a limitation of the objective function and therefore, needs to be specified as a constraint. Moreover, the constraint is required to specify the percentage increase in RMS error that results from the removal of each input parameter from the model. This constraint ensures that the cost associated with an increase in error due to the removal of input parameters, resulting in a decrease in input parameter measurement cost, is considered in the optimal solution.

iv). The decrease in computation time resulting from a decreased number of input parameters is a necessary constraint on the objective function. It is shown that computation time decreases with decreasing model complexity. The relationship that exists between computation time and the number of input parameters in a model must be incorporated into a suitable constraint.

v). The decision variables used to represent input parameters must be binary, ie. assigned a value of either 0 or 1 in the optimal solution.

vi). At least one input parameter must be assigned a value of 1 in the optimal solution. In order to develop a process model for each industrial application it is necessary to use at least one process parameter in order to predict the output value in each instance.

vii). A constraint specifying non-negativity is also necessary, as all decision variables used in the objective function are required to be positive.

While the listed constraints provide sufficient information to impose the required limitations on the objective function, the decision variables and the values used in the constraints are unique to each model. Therefore, it is necessary to specify an individual set of constraints for each of the applied models to represent the error-number of parameters-computation time relationship. This is a particularly arduous and time consuming task as there are 6 neural network models to select among for each industrial application and further, it has been shown that there are 7 constraints required for each model. Hence, it would be necessary to develop a total of 42 constraints for each application in order to select the most suitable model. Moreover,

for the 3 industrial applications studied as a part of this work, development of 126 constraints would be required. While the majority of constraints are relatively simple to develop, it is noted that constraints (iii) and (iv) are particularly difficult to specify due to the complex relationship that exists between RMS error, contributing input parameters and computation time. Hence, because constraint specification is arduous and time consuming for each application and further, there is no assurance that once the constraints are developed a suitable optimal solution will be provided by the simplex method, linear programming is considered an unsuitable optimisation technique for model selection. Consequently, it is necessary to develop a quantitative decision logic that allows efficient model selection by analysing and comparing all available options. While linear programming has been shown to be an unsuitable technique in this instance, the general framework for problem formulation associated with linear programming will be utilised in the developed optimisation strategy, which is introduced in the following chapter.

6.7 CONCLUDING REMARKS

While prediction accuracy is an important consideration when selecting a neural network for a specific application it is not satisfactory to select a particular neural network solely on the basis that it achieves minimum prediction error of the applied models. Rather, it is necessary to select the model that yields highest economic benefit when implemented. This involves consideration of the prediction error associated with the model, the number of parameters required and computation time associated with model development. Further, despite the diversity and broad range of neural network architectures and algorithms, all neural networks perform the same task; mapping input vectors to output vectors. However, each neural network differs in the way they perform this task. Hence, it follows that the combination of prediction error, number of parameters required and computation time associated with each model for a specific application will be different. Therefore, neural network selection for a specific application involves not only deciding on the combination of prediction error, number of parameters required and associated computation time for a particular neural network but also selecting from among many feasible alternative models. In order to make this complex decision it is necessary to apply an appropriate evaluation methodology. While particular operations research techniques have been investigated as a suitable optimisation strategy it is shown that

they are inefficient and unsuitable in this instance. However, it is important to note that the discussion of operations research and linear programming provided in this chapter was necessary to introduce some important concepts of the optimisation strategy that is introduced in the following chapter. In particular, while the solution strategies provided by linear programming are unsuitable in this instance, the technique associated with linear programming for problem formulation is a valuable methodology that will be incorporated into the developed optimisation technique.

CHAPTER SEVEN

Development and Application of Optimisation Technique

7.1 SPECIFICATION OF REQUIREMENTS OF OPTIMISATION TECHNIQUE

The main objective of the optimisation technique is to select from all available options a particular neural network model that is most suitable for an application and moreover, specify the most efficient and effective use of the selected model. Most suitable, in this instance, refers to the model that yields maximum economic benefit or the potential to maximise profitability, while the most efficient and effective use of the model is the strategy to achieve this. In order to identify the most suitable model for an application it is necessary to establish a methodology that allows comparison of the available alternatives. Further, the methodology must not be biased towards any particular model but rather, evaluate and compare all models equally. The strategy of problem formulation offered by the operations research techniques discussed in the previous chapter is an appropriate procedure to satisfy this requirement. Specifically, it is useful to establish appropriate decision variables for each application for model selection and further, structure the objective of the problem into an appropriate mathematical form. This allows the problem to be represented algebraically and analysed using precise mathematical techniques. Nevertheless, an essential requirement of the optimisation technique is that it must overcome the limitations of the existing operations research techniques introduced in the preceding chapter. In particular, the optimisation technique must be capable of efficiently specifying real and integer values in the optimal solution where appropriate. Further, the optimisation technique must eliminate the need to ascertain and specify unique constraints for each model and application. Rather, a fundamental requirement of the optimisation technique is the ability to analyse a neural network model based on the relationship each input parameter has with prediction error and

computation time. While this information is provided by the predictive importance analysis it is critical that the optimisation technique be capable of using the available information appropriately to determine an optimal solution.

7.1.1 Strategy for Problem Formulation

It has been noted that the problem formulation strategy associated with linear programming is suitable in this instance to specify the objective of the problem in an appropriate mathematical form. Hence, it is necessary to identify the decision variables associated with each application and represent them in terms of algebraic symbols. Further, it is necessary to identify the objective of the optimisation problem and represent it as a linear function of the decision variables.

Decision variables - Prediction error associated with each of the applied models, the process parameters used as potential neural network input variables and neural network computation time are each represented by a unique decision variable for each application. These particular decision variables represent each of the assessment criteria specified previously and are considered necessary and sufficient to make a decision of which neural network model to use for a specific application. Each decision variable is represented algebraically in the objective function as x_m, where $m = 1, \ldots, n$, where n is the total number of decision variables associated with the problem. It is important to note that while prediction error and computation time can each be expressed as a fraction, or real number, in the optimal solution, each decision variable that represents an input parameter must be expressed in the optimal solution as a whole number, or integer. This is due to the fact that an input parameter must be either included or excluded from the model; it is not possible to use part of a process parameter as an input variable. Moreover, a process parameter cannot be used more than once as an input variable to a particular neural network. Hence, the optimal solution must specify a value of either 0 or 1 for a decision variable that represents a process parameter.

Objective function - Total dollar cost is used as the measure of effectiveness for the objective function, as economic benefit is the primary and fundamental consideration for neural network selection in this instance. Further, the cost data associated with each of the decision variables is readily available and moreover, any cost data that is

not available can be established in a rational manner. In order to establish a mathematical form objective function for each industrial application it is necessary to determine the cost associated with each of the decision variables in each instance. In particular, it is necessary to determine the following:

i). cost per unit error - represents the cost associated with neural network prediction error

ii). cost per process parameter - represents the measurement cost associated with the process parameters used as potential neural network input variables

iii). cost per unit computation time - represents the cost associated with neural network training for model development

It is important to note that the objective function is written for each application to specify a minimisation problem in each instance. In particular, the objective function represents the prediction cost associated with each of the applied models and therefore must be minimised in order to achieve maximum economic benefit. Hence, the objective function is of the form:

$$\text{Minimise,} \qquad z = c_1x_1 + c_2x_2 + \ldots + c_mx_m \tag{7.1.1}$$

where, c_i = cost associated with decision variable i,

x_i = decision variable i, and

$i = 1, \ldots, m$

Specifically, the objective function represents mathematically the cost associated with a prediction using a particular neural network model. It incorporates the cost associated with prediction error, the cost associated with measurement of each of the process parameters used as neural network input variables and the cost associated with neural network computation time. Further, it can be seen that minimising the objective function involves establishing the particular combination of the decision variables that yields the minimum value of z.

7.2 ALGORITHM FOR THE DEVELOPED OPTIMISATION TECHNIQUE

While the criteria for model selection have been discussed in the previous chapter, it is useful to note that the complexity of model selection for a particular application is a function of the number of available models to choose from and the number of decision variables associated with the application. In particular, the complexity of the decision increases proportionally with an increasing number of models and decision variables. Moreover, there are many models and decision variables associated with each of the studied industrial applications. Hence, to allow for prompt and efficient optimisation analysis for multiple models and many decision variables and also to eliminate the need for manual computation it is necessary to develop a computer driven solution. In addition to a substantial saving of time and paperwork, the principal benefit of a computer driven solution is a program written specifically to determine an optimal solution for a neural network and moreover, select the most economically viable model to use for an application, rapidly on command. Further, development of a specific computer program to complete the analysis eliminates the potential for human error during the substantial number of calculations required during the optimisation analysis.

Hence, an optimisation technique executable on an electronic computer was developed for the purpose of neural network model selection. In particular, a program written specifically to determine an optimal solution for a neural network and moreover, select the most economically viable model to use for an application where maximising profitability is the main objective, was developed. Due to its combination of power and ease of use, Microsoft® Visual Basic™ [222-228] was selected as the programming language used to develop the optimisation technique. Visual Basic™ is an excellent tool for developing user friendly applications and is fully compatible with the popular Microsoft® Excel spreadsheet application, which is useful for displaying the optimisation analysis results in a user-friendly format and environment. Further, while necessary data handling and manipulation is easily and efficiently completed within the Microsoft® Excel environment, the broad range of spreadsheet functions available with Microsoft® Excel are utilised to interact with the user to obtain any required information for the optimisation analysis. Visual Basic™ communicates with Microsoft® Excel using object-oriented programming,

where, for example, a worksheet and cell are considered to be objects. In addition, all of the properties and processing techniques available with Microsoft® Excel can be controlled using Visual Basic™. Moreover, Visual Basic™ is a logical programming language that exploits common and useful programming techniques such as parameter passing, loop statements, function embedding and error handling.

While the development of an optimal solution for a specific neural network and subsequently, model selection for a particular application, are the main objectives of the developed computer driven solution, it has been noted in a previous chapter that further neural network analysis options are incorporated into the developed program. For instance, for a predetermined set of data patterns, the developed computer program can be used to manually or randomly select train and test data patterns and create the necessary neural network compatible text files for a predictive or casual importance analysis. Moreover, the developed program can be used to identify non-contributing input parameters in the neural network model and moreover, display the percentage contribution of each input parameter. While these features are useful for neural network analysis, they are critical for identifying an optimal solution for a particular neural network model and are therefore provided with the accompanying software. It has been noted that the developed program is named *Neural Network Analysis and Optimisation Strategy* and has the file name *NetAnal.xls*. The comprehensive user guide, attached as Appendix D, details each of the neural network analysis options available with the accompanying software. The methodology and characteristics of the developed optimisation technique are discussed in detail in the following section.

7.2.1 Methodology and Characteristics of Developed Optimisation Technique

The methodology used to determine an optimal solution for a neural network model is based on a *what-if analysis* of the 'original' solution. Prior to discussing this methodology it is necessary to define what is meant by the original solution. The original solution is a combination of the selection criteria specified in the preceding chapter. Specifically, it is a combination of RMS error, number of input parameters and computation time associated with a particular model. RMS error is established in the first instance by identifying the neural network architecture and algorithm that produce minimum error when all potential input parameters are used in the model.

Subsequently, eliminating non-contributing input parameters from the identified neural network and re-training yields the value of RMS error in the original solution. The number of parameters in the original solution is the total number of contributing input variables in the neural network model. That is, the number of input parameters required to achieve the minimum RMS error in the original solution. Computation time in the original solution is the time required to achieve the minimum RMS error in the original solution, with all non-contributing input parameters omitted. Hence, the original solution specifies the minimum RMS error that can be achieved by a neural network for a given train and test data set, the minimum number of process parameters required to achieve the minimum RMS error and the associated computation time. The value of RMS error, number of input parameters required and associated computation time in the original solution are obtained directly from the predictive importance analysis results. For example, an original solution for the WH neural network for the electrolyte additive prediction application would be equivalent to that determined from the predictive importance analysis results shown in Appendix C, Table C.7 and summarised in Table 7.2.1. It can be seen that the test RMS error achieved by the WH model with all non-contributing input variables omitted from the data sets is 0.1121, while the computation time required to achieve this is 281.02 seconds. Further, the contributing process parameters in the model, identified using the predictive importance analysis, are assigned a value of 1 in the original solution, while non-contributing parameters are assigned a value of 0. It can be seen that there are seven contributing process parameters in the WH neural network for the electrolyte additive prediction application, namely, target bath temperature, bath temperature, bath resistivity, AlF_3 addition (t-1), Na_2CO_3 addition (t-1), Na content and temperature reference.

TABLE 7.2.1. Original Solution for WH Neural Network for Electrolyte Additive Prediction Application

Parameters Specified in Predictive Importance Analysis		Original Solution
RMS error (test) - non-contributing inputs removed		0.1121
Process parameters -	target bath temperature	1
	bath temperature	1
	bath height	0
	bath resistivity	1
	emf	0
	AlF_3 addition (t-1)	1
	Na_2CO_3 addition (t-1)	1

cell power	0
cell age	0
F content	0
Na content	1
temperature reference	1
Computation time (s) - non-contributing inputs removed	281.02

Hence, the original solution represents a feasible solution for the neural network model that has been confirmed by the predictive importance analysis as achievable. In addition, unnecessary prediction costs are eliminated from the model due to the removal of non-contributing process parameters, resulting in reduced costs for process parameter measurement and associated computation time. Remembering that prediction error, each process parameter used as a potential neural network input variable and computation time are each represented by a unique decision variable and further, identifying the cost associated with each decision variable, the cost associated with the original solution can be calculated. In particular, assigning the x_m decision variables as; x_1 to represent RMS error, x_{n+1} , where n = 1, ..., i, and i = number of process parameters, to represent each process parameter and x_{n+2} to represent computation time, an objective function of the standard form shown in Equation 7.2.1 is established.

$$\text{Minimise,} \quad z = c_1x_1 + c_{n+1}x_{n+1} + \ldots + c_{n+2}x_{n+2} \tag{7.2.1}$$

where, c_1 = cost per unit RMS error,

c_{n+1} = measurement cost of process parameter n, and

c_{n+2} = cost per unit computation time

The value of the objective function, z, is calculated by substituting the relevant values of the decision variables from the original solution into the objective function. Assuming that some process parameters are identified as non-contributing input variables in a particular neural network and further, that the non-contributing input variables have an associated measurement cost, it follows that the original solution is an improvement on the initial feasible solution where all process parameters are included in the model. Hence, completing the predictive importance analysis and

removing non-contributing process parameters from the neural network is economically beneficial as it reduces the value of the objective function.

It is useful to note that while the predictive and casual importance analyses have shown similar resulting RMS error as a consequence of the exclusion or variation of input variables in the train and test data sets, respectively, the computation time resulting from the elimination of an input variable is only given by the predictive importance analysis. Moreover, the predictive importance analysis shows the direct increase in RMS error when an input variable is omitted from the train and test data sets, which is critical information for completing the optimisation analysis. Therefore, it is necessary to use the predictive importance analysis results to complete the optimisation analysis, rather than the casual importance analysis results. However, it has been shown that the casual importance results are useful to compliment the percentage contribution results for the input parameters calculated using the predictive importance results.

Having clarified the procedure used to establish the original solution it is now interesting to discuss the methodology whereby an optimal solution for a neural network is identified by completing a *what-if analysis* of the original solution. It is termed a *what-if analysis* because the developed optimisation technique investigates the opportunity; what if parameter n, where $n = 1, ..., i'$, and i' = number of contributing process parameters, is removed from the original solution; what is the economic benefit. In particular, it has been noted that it may be economically beneficial to eliminate one or more contributing process parameters from a neural network model. It has been shown in the predictive importance analyses completed that the removal of any contributing process parameter from a neural network model results in increased prediction error. On the other hand, it has also been shown that the removal of an input variable from a neural network model results in decreased computation time, due to reduced model complexity. Hence, the removal of a contributing process parameter from a neural network model has two implications. Firstly, it results in increased RMS error and secondly, it results in decreased computation time, for which there is typically an associated cost disadvantage and benefit, respectively. If the error associated with the prediction of a particular process parameter is high, process efficiency generally decreases as a consequence, hence,

resulting in an economic disadvantage. Conversely, lower computation time typically results in an economic benefit, generally due to the reduced demand for computer resources.

The cost associated with error due to the prediction of a particular process parameter can generally be assigned a dollar value, as can the computation time associated with model development. Defining the resulting cost of increased RMS error minus decreased computation time due to the removal of process parameter n from a neural network as Y_n, the decision of whether to remove a contributing parameter from a neural network model is given by the following:

$$\begin{pmatrix}\text{remove contributing parameter } n \\ \text{from the neural network model}\end{pmatrix} = \begin{cases}\text{YES, if } CP_n \geq Y_n \\ \text{NO, if } CP_n < Y_n\end{cases} \tag{7.2.2}$$

where, CP_n = measurement cost of contributing process parameter n,

$n = 1, \ldots, i'$, and

i' = total number of contributing process parameters

To determine Y_n for a particular application it is necessary to establish the cost associated with RMS error and computation time. Further, the percentage increase in RMS error and percentage decrease in computation time in the original solution resulting from the removal of parameter n from the neural network must be determined. Subsequently, the decrease in cost associated with decreased computation time is subtracted from the increase in cost associated with an increase in RMS error to determine the value of Y_n. While the cost associated with RMS error and computation time are established by considering the economics of the application, the percentage increase in RMS error and percentage decrease in computation time resulting from the removal of a contributing process parameter is determined from the predictive importance analysis. In addition, it is important to note that the removal of multiple contributing parameters from a neural network model results in an accumulated percentage increase in RMS error and decrease in computation time. For example, consider the predictive importance analysis results for the WH neural network for the electrolyte additive prediction application, as summarised in Table 7.2.2. It can be seen that each contributing process parameter

yields a unique percentage increase in RMS error and a percentage decrease in computation time when omitted from the train and test data sets. It is important to note that the percentage decrease in computation time resulting from the abstraction of a parameter from the data sets is theoretically the same for each parameter. That is, the removal of a process parameter from a neural network will yield a decrease in computation time that is common for all process parameters; it is simply reducing the complexity of the neural network by a single input. Further, the reduction in computation time is not influenced by the importance of the parameter. However, some variation in percentage decrease in computation time is documented in the results for each input variable, attributed to minor random variation in computer processing speed during neural network training. Therefore, an average percentage decrease in computation time is calculated and used for each input variable in the neural network model during the optimisation analysis. Hence, the total decrease in computation time due to the removal of multiple parameters from a neural network is calculated by multiplying the average percentage decrease in computation time by the number of input parameters omitted.

TABLE 7.2.2. WH Neural Network Predictive Importance (PI) Results for Electrolyte Additive Prediction Application

	RMS Error		Computation Time	
Input Variable Omitted	PI Value (Test)	Increase (%)	PI Value (s)	Decrease (%)
no variables omitted	0.1140	-	321.81	-
target bath temperature	0.1208	6.0	299.36	7.0
bath temperature	0.1160	1.8	298.92	7.1
bath height	0.1140	0.0	299.30	7.0
bath resistivity	0.1309	14.8	300.83	6.5
emf	0.1136	-0.4	299.15	7.0
AlF_3 addition (t-1)	0.1150	0.9	299.71	6.9
Na_2CO_3 addition (t-1)	0.1146	0.5	301.07	6.4
cell power	0.1132	-0.7	300.52	6.6
cell age	0.1137	-0.3	298.44	7.3
F content	0.1135	-0.4	298.22	7.3
Na content	0.1156	1.4	298.93	7.1
temperature reference	0.1196	4.9	300.36	6.7

It is important to note that the percentage contribution of a process parameter determined using the predictive importance analysis is not the same as the percentage increase in RMS error when the process parameter is omitted from the neural

network, due to the procedure used to calculate either value. While the procedure used to calculate the percentage contribution of an input parameter has been provided in a previous chapter, the percentage increase in RMS error resulting from the removal of a parameter from the original solution is calculated using the following formula:

$$err_{rms\ n} = \left(\frac{err_{rms\ (param\ n)} - err_{rms\ (orig)}}{err_{rms\ (orig)}} \right) \tag{7.2.3}$$

where, $err_{rms\ n}$ = percentage increase in RMS error due to removal of parameter n, where n = 1, ..., i',

i' = number of contributing input variables,

$err_{rms\ (orig)}$ = RMS error when no inputs are omitted, and

$err_{rms\ (param\ n)}$ = RMS error when parameter n is omitted

Likewise, the decrease in computation time resulting from the removal of an input parameter from the train and test data sets is calculated using the following formula:

$$tim_{comp\ n} = \left(\frac{tim_{comp\ (orig)} - \overline{tim_{comp\ (param\ n)}}}{tim_{comp\ (orig)}} \right) \tag{7.2.4}$$

where, $tim_{comp\ n}$ = percentage decrease in computation time due to removal of parameter n, where n = 1, ..., i',

i' = number of contributing input variables,

$tim_{comp\ (orig)}$ = computation time when no inputs are omitted, and

$\overline{tim_{comp\ (param\ n)}}$ = average computation time when parameter n is omitted

Referring to the predictive importance analysis results documented in Table 7.2.2, the accumulated percentage increase in RMS error due to the removal of multiple inputs, say target bath temperature (6.0%) and bath temperature (1.8%), would result in an accumulated percentage increase in RMS error of 7.8%. Hence, the accumulated error is simply the sum of the individual percentage increase in error for the multiple

inputs removed. This particular cumulative behaviour has been confirmed by removing multiple input parameters from the train and test data sets and re-training the developed neural network models to investigate the associated RMS error. In each instance, the resulting RMS error due to the removal of multiple input variables from a neural network was approximately equivalent to the original RMS error plus the original RMS error multiplied by the accumulated percentage increase in RMS error. Hence, the removal of target bath temperature and bath temperature from the WH neural network for electrolyte additive prediction would yield a test RMS error of 0.1229, (ie. 0.1140 + 0.1140 x 7.8%). Therefore, the increase in RMS error due to the removal of multiple input parameters from a neural network model is given by:

$$\begin{aligned} err_{rms\,(res)} &= err_{rms\,(orig)} + err_{rms\,(orig)} \times err_{(accum)} \\ &= err_{rms\,(orig)} (1 + err_{(accum)}) \end{aligned} \tag{7.2.5}$$

where, $err_{rms\,(res)}$ = resulting RMS error due to removal of multiple inputs,

$err_{rms\,(orig)}$ = RMS error when no inputs are omitted, and

$err_{(accum)}$ = accumulated percentage increase in RMS error due to removal of multiple parameters and is calculated using:

$$err_{(accum)} = \sum_{1}^{n'} err_{rms\,n} \tag{7.2.6}$$

where, n' = total number of contributing input variables removed

Similarly, the decrease in computation time due to the removal of multiple input parameters from a neural network model is given by:

$$\begin{aligned} tim_{comp\,(res)} &= tim_{comp\,(orig)} - tim_{comp\,(orig)} \times tim_{(accum)} \\ &= tim_{comp\,(orig)} (1 - tim_{(accum)}) \end{aligned} \tag{7.2.7}$$

where, $tim_{comp\,(res)}$ = resulting computation time due to removal of multiple parameters,

$tim_{comp\,(orig)}$ = computation time when no inputs are omitted, and

$tim_{(accum)}$ = accumulated percentage increase in computation time due

to removal of multiple parameters and is calculated using:

$$tim_{(accum)} = \sum_{1}^{n'} tim_{comp\,n} \tag{7.2.8}$$

where, n' = total number of contributing input variables removed

Hence, two fundamental concepts of the developed optimisation analysis are stated in the following:

i). the removal of a contributing process parameter from the original solution yields a unique increase in RMS error and a decrease in computation time, and

ii). the removal of multiple contributing process parameters from the original solution yields an accumulated increase in RMS error and accumulated decrease in computation time

In addition, while it is recognised that the costs associated with the decision variables do not remain constant due to a changing global economy and as a result of process advances, it is important to note that the decision variable costs are calculated at an instant in time for the purpose of the optimisation analysis. Hence, the optimisation analysis is considered to be a static problem. However, the decision variable costs used in the optimisation analysis are the most recent at the time of analysis. In addition, it is worthwhile noting that the relationship of the decision variable costs to the problem is deterministic. Specifically, the value of the objective function is determined by the values of the decision variable costs. Incorporating these fundamental concepts, the methodology of the developed optimisation technique to establish an optimal solution for a neural network is summarised in the following algorithm.

Step 1. Establish the total number of decision variables required for the application. Assign one decision variable to represent RMS error (x_1), a decision variable to represent each potential input parameter (x_{n+1}), where $n = 1, ..., i$ and i = number of potential input parameters, and one decision variable to represent computation time (x_{n+2}).

Step 2. Determine the cost associated with each decision variable; c_1 = cost per unit RMS error, c_{n+1} = measurement cost of process parameter n and c_{n+2} = cost per unit computation time. Represent the objective function for the application in the following mathematical form:

$$\text{Minimise,} \quad z = c_1x_1 + c_{n+1}x_{n+1} + \ldots + c_{n+2}x_{n+2} \tag{7.2.9}$$

Step 3. Identify the original solution for the neural network model studied. Hence, determine the minimum RMS error that can be achieved by the neural network for a given train and test data set, the minimum number of process parameters required to achieve the minimum RMS error and the associated computation time. This information is obtained directly from the predictive importance analysis results.

Step 4. Establish the cost associated with the original solution. This is achieved by substituting the values of the decision variables given in the original solution into the objective function given in Equation 7.2.9 to calculate z.

Step 5. Determine the percentage increase in test RMS error if contributing process parameter n is removed from the neural network model. This information is obtained from the predictive importance analysis results and is calculated using Equation 7.2.3.

Step 6. Determine the percentage decrease in computation time if contributing parameter n is removed from the neural network model. This information is obtained from the predictive importance analysis results and is calculated using Equation 7.2.4.

Step 7. For each n, calculate the cost increase associated with the error increase due to the removal of contributing parameter n. The cost associated with an increase in RMS error, denoted as E_n, is calculated using the formula:

$$E_n = \left(\frac{err_{rms\ (param\ n)} - err_{rms\ (orig)}}{err_{rms\ (orig)}} \right) . c_1 \tag{7.2.10}$$

Step 8. For each n, calculate the cost decrease associated with the computation time decrease due to the removal of contributing parameter n. The cost

associated with a decrease in computation time, denoted as C_n, is calculated using the formula:

$$C_n = \left(\frac{tim_{comp\ (orig)} - \overline{tim_{comp\ (param\ n)}}}{tim_{comp\ (orig)}} \right) \cdot c_{n+2} \qquad (7.2.11)$$

Step 9. For each n, subtract the cost associated with a decrease in computation time from the cost associated with an increase in error to determine the total cost due to the removal of parameter n. The resultant cost is denoted as Y_n and is calculated using the formula:

$$Y_n = E_n - C_n \qquad (7.2.12)$$

Step 10. For each n, establish whether the measurement cost of each contributing parameter n, denoted as CP_n, is greater-than-or-equal-to or less-than Y_n and consequently, assign a 0 value to any decision variable x_{n+1} for which the associated parameter measurement cost is greater-than-or-equal-to Y_n. Hence, for each n, the value of the decision variable x_{n+1} is determined using the following:

$$(x_{n+1}) = \begin{cases} 0, \text{ if } CP_n \geq Y_n \\ 1, \text{ if } CP_n < Y_n \end{cases} \qquad (7.2.13)$$

Step 11. Check that at least one contributing input parameter is assigned a value of 1 in the optimal solution, ie. $\Sigma x_{n+1} > 0$. If no contributing input parameter is assigned a value of 1 in the optimal solution then re-instate the contributing input parameter that saves the least money when removed from the model, ie. re-instate parameter n that yields minimum $CP_n - Y_n$.

Step 12. Calculate the resulting RMS error due to the removal of any contributing input parameters from the original solution. This is calculated using Equation 7.2.5. The resulting RMS error in the optimal solution becomes the value of the decision variable x_1 in the objective function.

Step 13. Calculate the resulting computation time due to the removal of any contributing input parameters from the original solution. This is calculated

using Equation 7.2.7. The resulting computation time in the optimal solution becomes the value of the decision variable x_{n+2} in the objective function.

Step 14. Establish the cost associated with the optimal solution. This is achieved by substituting the values of the decision variables given in the optimal solution into the objective function to calculate z.

To aid further understanding of the systematic procedure used to establish an optimal solution for a neural network, Figure 7.2.1 illustrates the methodology of the developed optimisation technique.

While the optimisation analysis incorporates specific routines to ensure that the removal of contributing process parameters from the original solution results in an appropriate increase and decrease in RMS error and computation time, respectively, it is necessary to confirm the optimal solution is achievable by re-training the associated neural network. In this instance, the neural network is trained and tested using only those process parameters that are assigned a value of 1 in the optimal solution. Consequently, the associated RMS error and computation time following convergence of the neural network model should be comparable to that specified in the optimal solution. It is important to note that a rigorous investigation was completed to incorporate appropriate procedures into the developed optimisation technique to increase RMS error and decrease computation time correctly and precisely due to the removal of specific process parameters from the original solution. However, there is the potential for one or more of the following to occur when the neural network is re-trained using only those parameters specified in the optimal solution:

i). RMS error after re-training the relevant neural network is lower than that specified in the optimal solution

ii). computation time after re-training the relevant neural network is lower than that specified in the optimal solution

iii). RMS error after re-training the relevant neural network is higher than that specified in the optimal solution

iv). computation time after re-training the relevant neural network is higher than that specified in the optimal solution

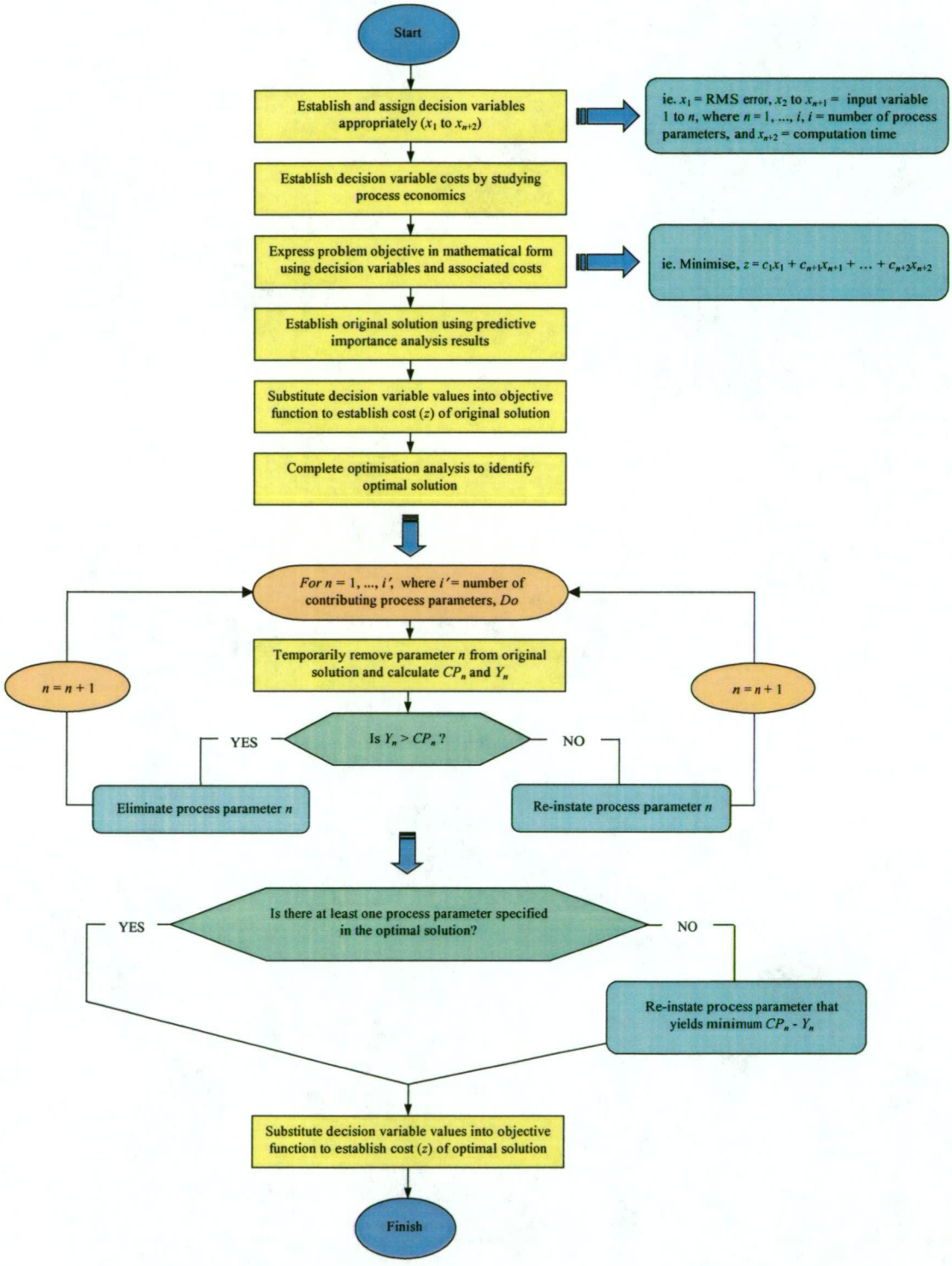


Fig. 7.2.1. Flow Chart Highlighting Methodology of Developed Optimisation Technique to Establish an Optimal Solution for a Neural Network

While it is noted that consequences (i) or (ii) may occur as a result of re-training the relevant neural network, it is expected that consequences (iii) or (iv) are more likely to occur. However, if consequences (i) and (ii) do occur as a result of re-training the neural network it is economically beneficial, as lower RMS error or computation time yield a lower value of z in the minimisation objective function. On the other hand, the occurrence of consequences (iii) or (iv) yield a higher value of z in the minimisation objective function and therefore are of economic disadvantage. If it is found that the RMS error specified in the optimal solution is not achievable using the relevant neural network, it is typically attributed to the fact that some of the process parameters used in the model are statistically dependent. Specifically, the importance of a particular input variable may be dependent on what other input parameters are included in the model. For example, consider the electrolyte additive prediction application and the predictive importance analysis results documented in Table 7.2.2. It is shown that the percentage increase in RMS error resulting from the removal of bath temperature is 1.8%. If, for instance, bath temperature is statistically related to, say, bath resistivity and bath resistivity is an input variable in the neural network model, then the percentage increase in error due to the removal of bath temperature may only be 1.8% due to the fact that the majority of the information that bath temperature provides in the model is already specified by bath resistivity. To emphasise this point, consider a reduction cell that has a high bath temperature and consequently, requires a high AlF_3 addition. In order to predict the amount of AlF_3 to schedule to the reduction cell it may not be necessary to know the bath temperature of the cell if the bath resistivity is known, as bath resistivity provides crucial information regarding bath chemistry. However, if bath resistivity is not an input variable in the neural network model for electrolyte additive prediction then it is expected that the importance of bath temperature would be significantly higher as a consequence. Hence, the removal of bath temperature and bath resistivity from the model may yield a higher accumulated percentage increase in RMS error than 16.6%, ie. bath temperature (1.8%) plus bath resistivity (14.8%). While it is possible to obtain specific information regarding the statistical dependence of process parameters during the predictive importance analysis, it is not an efficient procedure. It would be necessary to, rather than simply omit each input parameter individually from the data sets and train the neural network accordingly, omit all possible combinations of each process parameter and train the neural network accordingly.

Hence, for i input variables, a predictive importance analysis to determine the statistical dependence of process parameters would involve creating $i(i\text{-}1)$ train and test text files and training the neural network $i(i\text{-}1)$ times, compared to creating i train and test text files and training the neural network i times using the existing technique. For example, considering the electrolyte additive prediction application, for 12 input variables a predictive importance analysis to determine statistical dependence would involve creating 12 x (12-1), ie. 132, train and test text files and training the neural network 132 times, compared to creating 12 train and test text files and training the neural network 12 times using the existing technique. A more efficient process is to complete a standard predictive importance analysis and determine an optimal solution for a particular neural network. Consequently, re-train the neural network in the first instance using the process parameters included in the optimal solution in an endeavour to achieve comparable RMS error to that specified in the optimal solution. However, if this is not achievable due to the statistical dependence of some omitted parameters then it is necessary to re-instate particular process parameters until a feasible solution is obtained that yields a lower value of z in the minimisation objective function than that given by the original solution. Further, it is economically beneficial to re-instate those process parameters that yield the minimum value of $CP_n - Y_n$ until a feasible solution is accomplished. The feasible solution that yields the minimum value of z in the minimisation objective function is excepted as the optimal solution. Nevertheless, the original solution provides an upper-limit feasible solution that has been confirmed as being achievable using the relevant neural network model.

In regard to consequence (iv), minor discrepancies in the resulting computation time may occur, which are typically attributed to variations in computer processing speed. Such discrepancies are due to random variation and are therefore statistically insignificant. However, a higher number of iterations may be required to achieve convergence of the network weights, which may yield a higher computation time. Consequently, the value of z in the minimisation objective function may slightly increase as a result of a longer computation time. Notwithstanding, the original solution provides an upper-limit feasible solution that has been confirmed as being achievable using the relevant neural network model.

While the main objective of the developed optimisation technique is to establish an optimal solution for a neural network model, the accompanying software also incorporates relevant procedures to identify the best neural network to use for a particular application. The decision of which neural network model to use for a particular application is completed by calculating the cost associated with each optimal solution for the applied models and subsequently, selecting the model that yields minimum prediction cost. While this can be achieved by manual observation of the multiple optimal solutions established for a particular application, this procedure has been incorporated into the developed computer program to provide the option of rapid automatic neural network selection.

7.2.2 Appraisal of Developed Optimisation Technique

While the developed optimisation technique accomplishes the same objective as available operations research techniques, such as linear programming, the developed optimisation technique is a substantially more efficient procedure as it eliminates the requirement for the formal specification of constraints associated with the objective function. It is shown that the developed optimisation technique incorporates operations research techniques for problem formulation and subsequently, solves the formulated objective function using precise mathematical techniques. However, the requirement for constraint specification to place certain limitations on the objective function is not necessary with the developed optimisation technique as the required limitations are determined directly from the predictive importance analysis completed prior to the optimisation analysis. Hence, while there is no formal constraint specification procedure required for the developed optimisation technique, the objective function is not unconstrained. It is important to note that if the linear programming technique were used for the optimisation analysis it would still be necessary to complete a predictive importance analysis in order to establish the associated constraints. Hence, the requirement to complete a predictive importance analysis is not a disadvantage of the developed optimisation technique as it is necessary in either instance. However, the developed optimisation technique directly incorporates the results of the predictive importance analysis to place explicit limitations on the objective function, rather than an intermediate step whereby associated constraints are required to be established and specified prior to the optimisation analysis. This represents a considerable time saving as it has been

shown that formal constraint specification is an arduous and time-consuming task for neural network optimisation for the studied industrial applications.

In addition to eliminating the requirement for constraint specification, the developed optimisation technique is also capable of specifying real and integer values in the optimal solution, as required. The ability to specify integer values appropriately in the optimal solution has been identified as a limitation of the simplex technique, while the integer programming technique to achieve this has been highlighted as extremely inefficient. Nevertheless, the developed optimisation technique incorporates specific procedures to ensure real and integer values are appropriately specified in the optimal solution.

Moreover, the developed optimisation technique is an investigative and analytical program that incorporates systematic procedures to efficiently and accurately determine the optimum combination of RMS error, number of process parameters required and computation time to minimise the associated cost of process modelling. Consequently, the developed optimisation technique is efficient; it analyses the original feasible solution rapidly on command to investigate the potential of an improved feasible solution. However, it is noted that in some instances it may not be possible to reduce the value of the objective function beyond the original solution, consequently, the original solution becomes the optimal solution. In order to apply the developed optimisation technique and ultimately select an optimum model for each of the studied industrial applications it is necessary to establish the appropriate decision variables and objective function for each particular application.

7.3 ESTABLISHMENT OF DECISION VARIABLES AND OBJECTIVE FUNCTION FOR STUDIED INDUSTRIAL APPLICATIONS

The optimisation technique detailed in the previous section will be applied to each of the studied industrial applications to determine an optimal solution for each of the applied neural networks and moreover, select the most suitable model to use for each application. However, prior to this it is necessary to develop an objective function for each application, in mathematical form. This is achieved by representing RMS error, each potential input parameter and computation time algebraically using appropriate decision variables and further, establishing the unit cost of each decision variable.

The decision variables and associated costs are then formulated into an objective function for each of the three studied industrial applications, as shown in the following section.

7.3.1 Decision Variables and Objective Function for Electrolyte Additive Prediction Application

While it has been shown previously, it is useful here to highlight again the input and output variables used for this particular application, as shown in Figure 7.3.1 using an arbitrary neural network model.

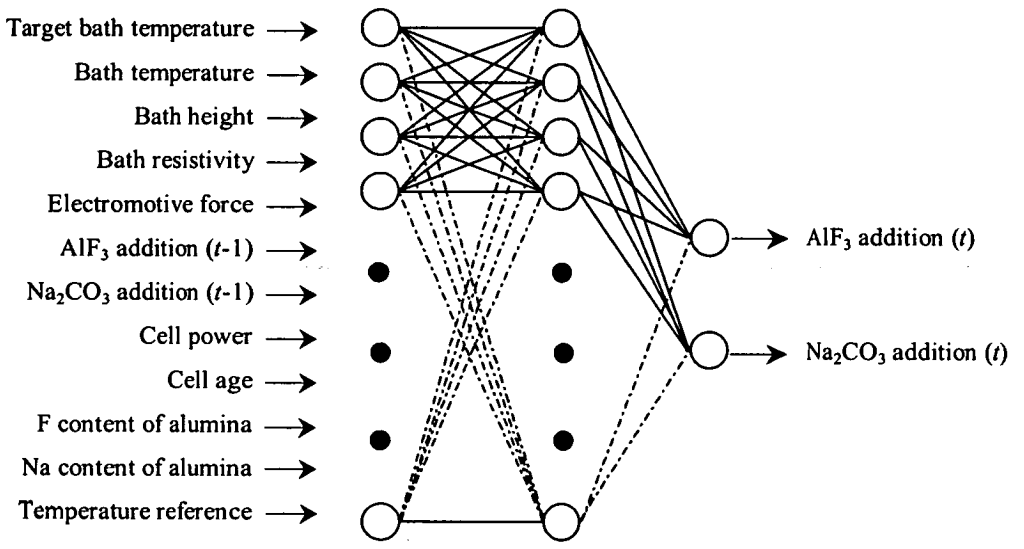


Fig. 7.3.1. Illustration of Neural Network Model for Electrolyte Additive Prediction Application

The x_{n+2} decision variables for the electrolyte additive prediction application are assigned to RMS error, each process parameter and computation time, as shown in Table 7.3.1. It is noted that as there are 12 process parameters used as potential input variables, there are 14 decision variables required for this particular application.

TABLE 7.3.1. Allocation of Decision Variables for Electrolyte Additive Prediction Application

		Decision Variable
Test RMS error (electrolyte additive prediction)		x_1
Process parameters -	target bath temperature	x_2
	bath temperature	x_3
	bath height	x_4
	bath resistivity	x_5

emf	x_6
AlF_3 addition (t-1)	x_7
Na_2CO_3 addition (t-1)	x_8
cell power	x_9
cell age	x_{10}
F content	x_{11}
Na content	x_{12}
temperature reference	x_{13}
Computation time required for model convergence	x_{14}

Cost per unit error - It is shown that the output variables for the electrolyte additive prediction application represent the quantity of AlF_3 and Na_2CO_3 to add to the reduction cell. Consequently, prediction error associated with the neural network models for this application directly results in incorrect quantities of AlF_3 and Na_2CO_3 being added to the reduction cell. Further, it is useful to note that the output variables for this application, while normalised and therefore dimensionless in the data sets used for neural network training and testing, were specified in units of kilograms prior to normalising. Hence, prediction error associated with the neural network models directly results in incorrect addition quantities of AlF_3 and Na_2CO_3 in kilogram units. However, the neural network has shown no significant bias towards predominantly higher-than-actual or lower-than-actual predictions of AlF_3 or Na_2CO_3. This is confirmed by the error distribution and analysis documented in Chapter Five which has shown that the mean error for AlF_3 and Na_2CO_3 predictions is approximately 0.21 and 0.48kg, respectively, which is considered to be negligible for the purpose of this analysis. Hence, additions of these important electrolyte additives will be approximately equally higher-than-actual as they are lower-than-actual. Hence, the quantity of additives wasted due to higher-than-actual predictions will be approximately equal to the quantity of additives saved due to lower-than-actual predictions. Hence, AlF_3 and Na_2CO_3 wastage is considered negligible and therefore not factored into the error cost for the purpose of this analysis. Nevertheless, incorrect additions of these important electrolyte additives can have an adverse effect on cell stability. In particular, the thermal balance of a reduction cell is significantly influenced by electrolyte chemistry. It has been shown that incorrect quantities of chemicals in the electrolyte results in low current efficiency. Moreover, it is shown in Figure 7.3.2 that the quantity of aluminium produced by a reduction cell is significantly influenced by the current efficiency of the cell. The high

correlation between current efficiency and aluminium production is confirmed by the correlation coefficient, *r*, of 0.909, as noted in Figure 7.3.2.

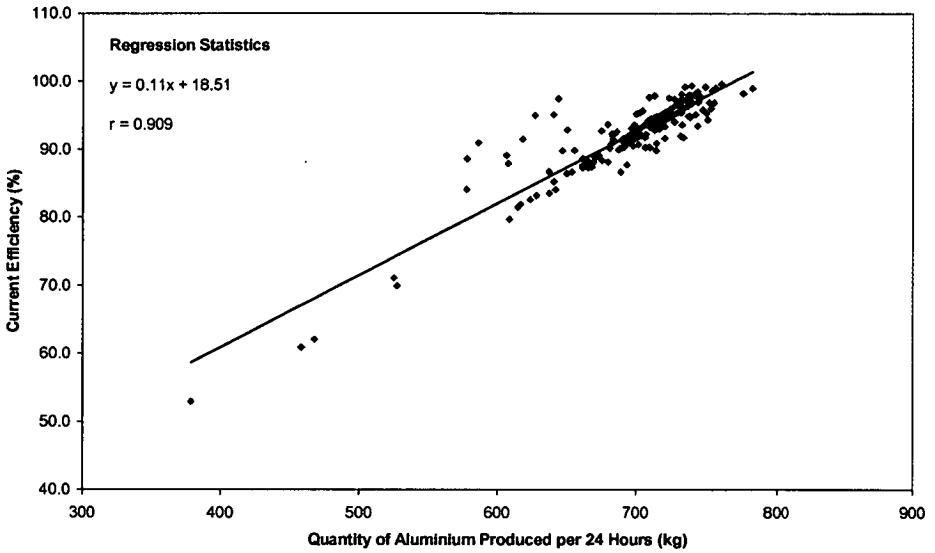


Fig. 7.3.2. Correlation Between Current Efficiency and Aluminium Production Rate per Twenty-Four Hour Period at CABBL

Observation of the graph shows that a current efficiency of 100.0% results in aluminium production at a rate of approximately 768.0kg per 24 hour period, while a 1.0% decrease in current efficiency results in a decrease of approximately 9.2kg of aluminium produced per 24 hour period. This is given by the inverse of the coefficient of the regression line equation. Hence, 80.0% current efficiency corresponds to an aluminium production rate of approximately 584.0kg per 24 hour period.

In order to determine the relationship between electrolyte chemistry and aluminium production rate it is necessary to consider the decrease in current efficiency with AlF_3 and Na_2CO_3 deviation from target. Hence, for a nominal electrolyte mass in the reduction cell of 1800.0kg, a 1.0kg addition of AlF_3 or Na_2CO_3 results in an increase of approximately 0.06%wt of the corresponding additive. Further, it has been shown previously for AlF_3 that a 1.0%wt deviation from target results in a decrease in current efficiency of 1.66%. As Na_2CO_3 has a stoichiometric ratio of approximately 1.0 with AlF_3 then the equivalent applies for Na_2CO_3. Hence, a 1.0%wt deviation of Na_2CO_3 from target results in a decrease in current efficiency of 1.66%.

Consequently, a deviation of 0.06%wt of AlF_3 and Na_2CO_3 results in a decrease of 0.10% current efficiency, corresponding to a decreased aluminium production rate of 0.92kg. The cost of this reduced aluminium production rate is determined by considering the cost associated with the production of a unit of aluminium, which is approximately $920.00 per tonne at CABBL, and the sale price of a unit of aluminium, which is set by the London Metal Exchange (LME) and is at a rate of approximately $2,650.00 per tonne [229]. Hence, the revenue on aluminium is at a rate of $1,730.00 per tonne, ie. sale price ($2,650.00) - production cost ($920.00), or alternatively, $1.73 per kilogram. Therefore, a 0.92kg reduction in aluminium production per 24 hour period results in lost revenue of $1.59 for the same period. Hence, the cost associated with 1.0kg deviation of AlF_3 and Na_2CO_3 from target is at a rate of $1.59.

In order to incorporate this coefficient into the particular objective function developed for the electrolyte additive prediction application it is necessary to determine error cost in terms of a unit of normalised RMS error. This is necessary because the train and test data sets are normalised prior to neural network training and consequently the error value calculated for a neural network is normalised RMS error. Hence, it is useful to have the error cost in an equivalent term as that associated with the neural network so that the neural network error value can be directly substituted into the objective function without the need for prior processing. Therefore, it is necessary in this instance to determine the RMS error equivalent of a prediction error of 1.0kg. While it has been shown previously, it is useful to note here that a value, x, is normalised using the following equation:

$$(x_i)_{norm} = \frac{x_i - \min(x_j)}{\max(x_j) - \min(x_j)} \tag{7.3.1}$$

where, $(x_i)_{norm}$ = normalised *ith* value in a set of j values,

x_i = original i^{th} value in a set of j values,

$\min(x_j)$ = original minimum value in a set of j values, and

$\max(x_j)$ = original maximum value in a set of j values

Further, the difference between a predicted value of x_i, $pred(x_i)$, and a target value of x_i, $targ(x_i)$, is given by:

$$err(x_i) = pred(x_i) - targ(x_i) \tag{7.3.2}$$

Now, substituting Equation 7.3.1 into Equation 7.3.2 yields the following relationship:

$$\begin{aligned} err(x_i)_{norm} &= pred(x_i)_{norm} - targ(x_i)_{norm} \\ &= \frac{pred(x_i) - \min(x_j)}{\max(x_j) - \min(x_j)} - \frac{targ(x_i) - \min(x_j)}{\max(x_j) - \min(x_j)} \\ &= \frac{pred(x_i) - targ(x_i)}{\max(x_j) - \min(x_j)} \\ &= \frac{err(x_i)}{\max(x_j) - \min(x_j)} \end{aligned} \tag{7.3.3}$$

In addition, Equation 7.3.3 is substituted into Equation 2.3.7 to yield a formula that can be used to determine the normalised RMS error, $err_{rms\ (norm)}$, equivalent of the error associated with the original units of the output variable(s), regardless of the application. This formula is derived as follows:

$$\begin{aligned} err_{rms\ (norm)} &= \sqrt{\frac{\sum_p \sum_k \left(err(x_i)_{norm}\right)^2}{n_p n_k}} \\ &= \sqrt{\frac{\sum_p \sum_k \left(\frac{err(x_i)}{\max(x_j) - \min(x_j)}\right)^2}{n_p n_k}} \end{aligned} \tag{7.3.4}$$

In this instance, the formula given in Equation 7.3.4 is used to determine the RMS error equivalent of 1.0kg error. For the electrolyte additive prediction application, the output parameters were normalised using $\min(x_j)$ equal to 0.0kg and $\max(x_j)$ equal to 100.0kg, for both output nodes. For n_k equal to 2 and n_p equal to 1,365, using the training data set, or alternatively, n_p equal to 200, using the test data set, and setting $err(x_i)$ equal to 1.0kg, substituting these values into Equation 7.3.4 yields a prediction

error of 1.0kg as equivalent to a normalised RMS error value of 0.01. Recalling that 1.0kg error has an associated cost of $1.59, then it follows that a RMS error value of 1.0 has an associated cost of $159.00. Hence, a unit of RMS error for the electrolyte additive prediction application has an associated cost of $159.00. Consequently, the coefficient of the decision variable x_1 in the objective function is 159.00.

Cost per process parameter - The cost associated with the measurement of each process parameter used as a potential input variable in the neural network models applied to this particular application are given in Appendix E, Table E.1. It is useful to note that the measurement cost of each process parameter was obtained by retrieving appropriate information from the smelter knowledge base. In particular, for each process parameter an investigation was completed to identify the associated measurement equipment replacement and maintenance costs. Further, for manually measured parameters a time study was completed to identify the time required to complete a measurement of the respective parameter. Subsequently, the labour required to complete the parameter measurement was converted to an appropriate cost in each instance. It is important to note that the measurement costs documented in Table E.1 represent the cost of a single measurement of the respective parameter.

It can be seen from the measurement costs that process parameters requiring manual measurement typically have a higher associated cost than those that are measured using automated procedures. It is useful to note that the measurement techniques used to acquire data for each of the process parameters used as potential input variables for this particular application are discussed in Chapter Four. While each manual measurement procedure requires a certain labour content, equipment depreciation is also typically higher when associated with a manual measurement procedure, compared to an automated procedure. Notwithstanding, process parameters that are measured and recorded using an automated procedure, such as bath resistivity and low and high frequency noise, also attract a measurement cost due to associated equipment depreciation and maintenance. On the other hand, some of the input variables listed in the table are shown to have a measurement cost of $0.00. This is attributed to the fact that the corresponding input variables are a specified value or calculated using appropriate formulae and are not physically measured, such as target bath temperature and cell age. The procedures used to derive

values for such parameters have negligible cost. Where possible, the measurement cost of a process parameter is shown in components of labour, equipment and associated injuries, if applicable. The values of the respective input variables form the coefficients of the decision variables x_2 to x_{13} in the objective function for this application.

Cost per unit computation time - Cost per unit computation time is calculated by regarding computer time as a resource, noting that neural network training is completed for each of the studied industrial applications using an electronic digital computer. In particular, the cost associated with running a computer per unit time is calculated. Neural network training considered here assumes the optimum neural network architecture determined previously for each of the studied neural network models will not change after implementation, only the weights of the neural network will require periodic updating. In addition, data preparation for training is not considered here as it is the same for each neural network model, it is the computation time required to learn this data that is significant. Hence, human labour required to complete neural network training is considered negligible when costing neural network computation time. Therefore, the information required for this calculation includes electricity cost per unit time and computer power usage per unit time. For convenience, a single unit of time is defined as one hour (1.0hr) in this instance.

Electricity cost is at a standard rate of \$0.74E-01 per kilowatt-hour (kWhr) [230], while a typical computer used for neural network training has a power rating of 900 watts, or 0.900 kW. Hence, neural network computation time has an associated cost of \$0.74E-01 per kWhr, multiplied by 0.900 kW equates to \$0.67E-01 per hour, representing the cost per hour per training session. However, the cost associated with the decision variables representing RMS error and process parameters in the objective function are expressed in terms of cost per prediction. Hence, for consistency of the objective function it is necessary to calculate computation time on a cost per prediction basis. This is achieved by considering the number of predictions the neural network will make per year, which is at a rate of one prediction per day, therefore totalling approximately 365 predictions per year, and the number of training sessions required per year, estimated to be quarterly, ie. 4 training sessions per year, to update the neural network weights to account for process improvements and

changes. Hence, approximately 92 predictions are completed between training sessions. Therefore, the cost associated with computation time per prediction is at a rate of \$0.67E-01 per hour divided by 92, which equates to \$0.70E-03 per hour on a per prediction basis. While this value is relatively small and expected to have minor influence in the objective function it is included here for thoroughness of the optimisation analysis.

The decision variables and their relevant coefficients are now combined to formulate an objective function for this application, written in standard form as follows:

$$\text{Minimise,} \quad z = 159.00x_1 + 0.00x_2 + 1.15x_3 + 0.55x_4 + 0.02x_5 + 0.29x_6 + 1.17x_7 + 1.17x_8 + 0.04x_9 + 0.00x_{10} + 5.03x_{11} + 5.03x_{12} + 0.00x_{13} + 0.70\text{E-}03x_{14} \tag{7.3.5}$$

Hence, the objective function is a mathematical relationship that represents the cost associated with RMS error, each process parameter and computation time in a unique format for the electrolyte additive prediction application. It is important to note that this objective function is used to determine an optimal solution for each of the neural networks applied to the electrolyte additive prediction application. It can be seen that the objective function incorporates all process parameters that may be potential neural network input variables. While some neural network models may not require all of the process parameters present in the objective function, it is necessary to assign a decision variable to each parameter in this instance in order to make the objective function universal for all the applied neural networks. Specifically, the developed objective function is generic for this particular application, it is not necessary to determine an objective function for each neural network. However, the relationship between the process parameters and neural network error and computation time is unique to each neural network. Nevertheless, it has been noted that this information is easily determined directly from the predictive importance analysis completed for each applied neural network.

Further, it is useful to note that the resulting z value represents the cost associated with a prediction using the neural network model for which the particular optimisation analysis is completed. It incorporates an error cost, due to reduced

process efficiency as a consequence of incorrect additions of AlF_3 and Na_2CO_3, measurement cost of each process parameter used in the model and an allowance for computation time required to periodically re-train the neural network due to process changes and advances.

7.3.2 Decision Variables and Objective Function for Cell Failure Prediction Application

The process parameters used as potential input variables for neural network modelling for cell failure prediction are given in Figure 7.3.3, using an arbitrary neural network architecture.

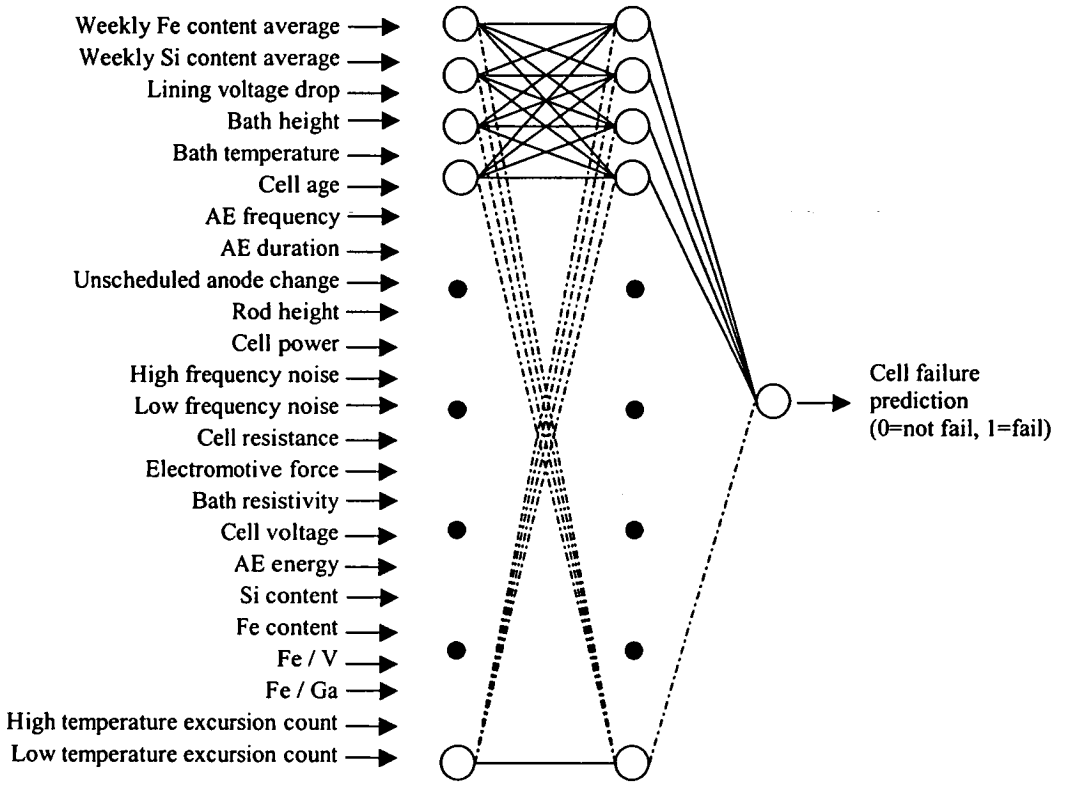


Fig. 7.3.3. Illustration of Neural Network Model for Cell Failure Prediction Application

The x_{n+2} decision variables associated with the cell failure prediction application are assigned to RMS error, each process parameter and computation time, as shown in Table 7.3.2. Hence, as there are 24 process parameters used as potential neural network input variables there are 26 decision variables required for this particular application.

TABLE 7.3.2. Allocation of Decision Variables for Cell Failure Prediction Application

		Decision Variable
Test RMS error (cell failure prediction)		x_1
Process parameters -	Fe content (weekly)	x_2
	Si content (weekly)	x_3
	lining voltage drop	x_4
	bath height	x_5
	bath temperature	x_6
	cell age	x_7
	AE frequency	x_8
	AE duration	x_9
	unscheduled anode change	x_{10}
	rod height	x_{11}
	cell power	x_{12}
	high frequency noise	x_{13}
	low frequency noise	x_{14}
	cell resistance	x_{15}
	emf	x_{16}
	bath resistivity	x_{17}
	cell voltage	x_{18}
	AE energy	x_{19}
	Si content	x_{20}
	Fe content	x_{21}
	Fe/V	x_{22}
	Fe/Ga	x_{23}
	high temperature count	x_{24}
	low temperature count	x_{25}
Computation time required for model convergence		x_{26}

Cost per unit error - It is noted that the output variable for the cell failure prediction application represents a not failure or failure condition, either 0 or 1, respectively. Moreover, prediction error in this instance results in the removal of a reduction cell from production, as a consequence of a predicted value of 1, or a cell remaining in production, as a consequence of a predicted value of 0. It is useful to restate that an output of 1 is produced for a cell that is predicted to tap-out while an output value of 0 represents a cell that is predicted to continue normal production, hence, not tap-out. In addition, a cell that is in fact going to soon tap-out but has a predicted value of 0 results in non-removal of the cell and consequently, a tap-out occurrence. On the other hand, a cell that is predicted to tap-out and consequently removed from production, but in fact would not tap-out if left in production, results in lost metal production from the removed reduction cell. Hence, error cost in this instance is a

combination of tap-out occurrences and lost aluminium production. The economic cost of a tap-out is a combination of lost production costs, material damage and labour costs associated with maintenance and necessary repair work. A typical value for each of these costs is given in the following:

Lost production:	$1,500.00	
Material damage:	$ 700.00	
Labour:	$ 300.00	
	$2,500.00	TOTAL

Hence, the total cost associated with a tap-out at CABBL is shown to be $2,500.00 on average. On the other hand, it is important to note that there is an economic cost associated with removing a reduction cell from production prior to failure. In particular, the economic cost associated with unnecessarily removing a reduction cell from production is due to unnecessary cell construction costs, calculated to be at a rate of approximately $34.00 per day. This means that if a reduction cell remains in production rather than removing it because it may tap-out, the cost saving equates to approximately $34.00 for each additional day the cell remains in production and does not tap-out. Hence, the cost of a tap-out at CABBL is approximately $2,500.00, while removing a cell from production unnecessarily results in lost production at a cost of approximately $34.00 per 24 hour period. In order to determine the value of the coefficient of the decision variable x_1 it is necessary to determine the number of non-predicted tap-out occurrences and days of lost production associated with a unit of RMS error. This is achieved by extrapolating a graph of RMS error versus non-identified tap-out occurrences and lost production. The number of non-identified tap-out occurrences and days of lost production associated with a particular RMS error are determined from the neural network training and test results. In particular, for each neural network applied to the cell failure prediction application and for a particular RMS error value associated with each neural network, the number of non-identified tap-out occurrences and days of lost production are calculated from the neural network training and testing results and plotted on a graph. Hence, for each network, the number of non-identified tap-out occurrences and days of lost production and the corresponding RMS error value are known. An average value representing the number of non-identified tap-out occurrences and days of lost

production per prediction for an RMS error value of 1.0 is obtained by extrapolating from the known data points, as shown in Figure 7.3.3. The extrapolated values represent an average number of non-identified tap-out occurrences and days of lost production that are expected for a neural network with a RMS error value of 1.0. It can be seen for a RMS error value of 1.0 the corresponding number of non-identified tap-out occurrences and days of lost production per prediction are 0.18 and 1.52, respectively.

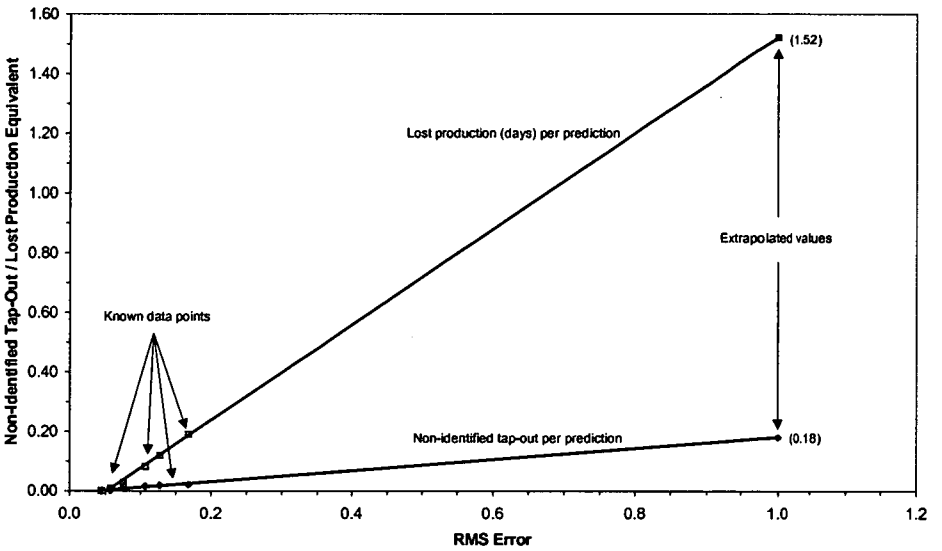


Fig. 7.3.3. RMS Error Equivalent of Non-Identified Tap-Out Occurrence and Lost Production (Days) per Prediction

Therefore, representing a non-predicted tap-out occurrence as u and lost production as v, the cost associated with a RMS error value of 1.0 is calculated as follows:

$$
\begin{aligned}
\text{1.0 unit RMS error} &= (u \text{ tap-outs} + v \text{ days lost production}) \text{ per prediction} \\
&= (u \times \$2{,}500.00 + v \times \$34.00) \text{ per prediction} \\
&= (0.18 \times \$2{,}500.00 + 1.52 \times \$34.00) \text{ per prediction} \\
&= \$501.68 \text{ per prediction}
\end{aligned}
$$

Hence, the value of the coefficient of the decision variable x_1 in the objective function is 501.68.

Cost per process parameter - The cost associated with the measurement of the process parameters used as input variables in the neural networks applied to this particular application are given in Appendix E, Table E.1. Similar to the previous application, process parameters requiring manual measurement typically have a higher associated measurement cost than those that are measured using automated techniques. Further, some parameters are assigned a measurement cost of \$0.00, corresponding to variables that are a specified value or calculated using appropriate formulae and are not physically measured. Similar to the previous application, the documented measurement costs represent the cost of a single measurement of the respective process parameter. The values of the respective input variables form the coefficients of the decision variables x_2 to x_{25} in the objective function.

Cost per unit computation time - It has been shown that the cost associated with neural network training is at a rate of \$0.67E-01 per hour. In order to calculate the cost per prediction it is necessary to divide this figure by the number of predictions each training session will be valid for. Similar to the previous application, neural network training will be completed quarterly throughout the year to account for process improvements and changes, while the number of predictions the neural network will complete is at a rate of 52 per year, ie. one prediction per week. Hence, each training session is valid for 13 predictions, equating to a computation time cost of \$0.51E-02 per hour.

The objective function for this particular application is written in standard form in Equation 7.3.6. It incorporates the unique combination of RMS error cost, process parameter measurement cost and the associated cost of computation time per prediction.

$$\begin{aligned}\text{Minimise,} \quad z = {} & 501.68x_1 + 3.83x_2 + 3.83x_3 + 0.68x_4 + 0.55x_5 + 1.15x_6 + 0.00x_7 + \\ & 0.02x_8 + 0.02x_9 + 0.26x_{10} + 0.61x_{11} + 0.04x_{12} + 0.01x_{13} + 0.01x_{14} + \\ & 0.03x_{15} + 0.29x_{16} + 0.02x_{17} + 0.01x_{18} + 0.02x_{19} + 3.83x_{20} + 3.83x_{21} \\ & + 3.83x_{22} + 3.83x_{23} + 1.15x_{24} + 1.15x_{25} + 0.51\text{E-}02x_{26} \qquad (7.3.6)\end{aligned}$$

While this objective function is generic for each of the neural networks applied to the cell failure prediction application, the unique relationship between each process

parameter and error and computation time is readily provided by the predictive importance analysis completed for each applied neural network.

7.3.3 Decision Variables and Objective Function for Electrolyte Temperature Prediction Application

It is shown in Figure 7.3.4, using an arbitrary neural network model, that electrolyte temperature is the single output for this application, while there are 10 process parameters specified as potential input variables.

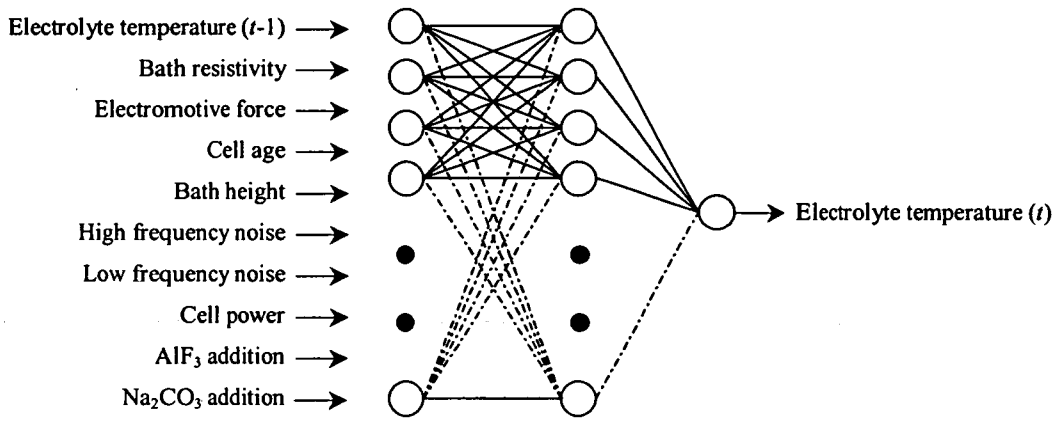


Fig. 7.3.4. Illustration of Neural Network Model for Electrolyte Temperature Prediction Application

The x_{n+2} decision variables for the electrolyte temperature prediction application are assigned to RMS error, each process parameter and computation time, as shown in Table 7.3.3. Hence, as there are 10 process parameters used as potential input variables for this application, there are 12 decision variables required.

TABLE 7.3.3. Allocation of Decision Variables for Electrolyte Temperature Prediction Application

		Decision Variable
Test RMS error (electrolyte temperature prediction)		x_1
Process parameters -	bath temperature (t-1)	x_2
	bath resistivity	x_3
	emf	x_4
	cell age	x_5
	bath height	x_6
	high frequency noise	x_7
	low frequency noise	x_8
	cell power	x_9

AlF_3 addition	x_{10}
Na_2CO_3 addition	x_{11}
Computation time required for model convergence	x_{12}

Cost per unit error - It is useful to initially select 1.0°C as a unit of error for this particular application, as the output variable is specified in units of °C prior to normalising, subsequently, deriving the equivalent units of normalised RMS error in order to determine the corresponding coefficient for the decision variable x_1. In order to determine the cost per unit error for this application it is necessary to note that the error associated with the prediction of electrolyte temperature directly influences the quantity of electrolyte additives scheduled to the reduction cell and consequently, influences the current efficiency of the cell. However, as the neural network results have shown no bias towards predominantly higher-than-actual or lower-than-actual predictions then the average electrolyte additive wastage is zero. This is confirmed by the error distribution and analysis documented in Chapter Five which has shown that the mean error for electrolyte temperature prediction is approximately -0.03°C, which is negligible. Thus, the cost associated with prediction error in this instance is due to inefficient operation of the reduction cell as a result of decreased current efficiency and does not incorporate electrolyte additive wastage.

It is shown in Figure 4.2.1 that a 1.0%wt change in AlF_3 yields a change in electrolyte temperature of 4.03°C, given by the coefficient of the regression line equation, or alternatively, a 1.0°C change in electrolyte temperature corresponds to a change in AlF_3 content of 0.25%wt. Further, for a nominal electrolyte mass of 1800.0kg, a 1.0°C change in electrolyte temperature results from the addition of 4.5kg of AlF_3, as 4.5kg is 0.25% of 1800.0kg. Similarly, due a stoichiometric ratio of unity, the same applies for Na_2CO_3. Moreover, it is useful to note that 1.0°C deviation from target corresponds to a decrease in current efficiency of 0.45%. Further, it has been shown that a decrease in current efficiency of 1.0% corresponds to a decrease in aluminium production at a rate of 9.2kg per 24 hour period. Hence, a current efficiency decrease of 0.45% corresponds to a decrease in aluminium production at a rate of 4.1kg per 24 hour period, at a cost of $1.73 per kilogram, equating to a cost of $7.09 per 1.0°C. Hence, the total cost associated with 1.0°C prediction error is $7.09. Using Equation 7.3.4 with $\min(x_j)$ equal to 930.0°C,

$\max(x_j)$ equal to 1,030.0°C, n_k equal to 1 and n_p equal to 1,244, using the training data set, or alternatively, n_p equal to 200, using the test data set, and setting $err(x_i)$ equal to 1.0°C, substitution yields 1.0°C error as equivalent to a normalised RMS error value of 0.01. Hence, the cost associated with a RMS error value of 1.0 is equal to $709.00. Consequently, the coefficient of the decision variable x_1 in the objective function is 709.00.

Cost per process parameter - The cost associated with measurement of the process parameters used as input variables in the neural network models applied to this particular application are given in Appendix E, Table E.1. The documented measurement costs represent the cost of a single measurement of the respective process parameter. Similar to the previous applications, process parameters requiring manual measurement typically have a higher associated measurement cost than those that are measured using automated techniques. Further, some input variables are assigned a measurement cost of $0.00 as they are variables that are a specified value or calculated using appropriate formulae and are not physically measured. The values of the respective input variables form the coefficients of the decision variables x_2 to x_{11} in the objective function for this application

Cost per unit computation time - Similar to the previous applications, it is necessary to determine the cost per unit computation time per prediction. While it has been shown that the cost associated with neural network training is at a rate of $0.67E-01 per hour, the number of neural network predictions is at a rate of approximately 365 per year for this particular application, ie. one prediction per day. Similar to the previous applications, neural network training will be completed quarterly throughout the year to account for process improvements and changes. Hence, each training session is valid for approximately 92 predictions, equating to a computation time cost of $0.70E-03 per hour per prediction.

Hence, the objective function in this instance incorporates an error cost, due to reduced process efficiency as a consequence of incorrect electrolyte temperature prediction, measurement cost of each process parameter used in the model and an allowance for computation time required to periodically re-train the model due to process changes and advances. It is written in standard form as follows:

$$\text{Minimise,} \quad z = 709.00x_1 + 1.15x_2 + 0.02x_3 + 0.29x_4 + 0.00x_5 + 0.55x_6 + 0.01x_7 + 0.01x_8 + 0.04x_9 + 1.17x_{10} + 1.17x_{11} + 0.70\text{E-}03x_{12} \quad (7.3.7)$$

Likewise to the previous applications the objective function specified in Equation 7.3.7 is generic for all neural networks applied for electrolyte temperature prediction. Specifically, the objective of the optimisation analysis completed for each neural network is to minimise this objective function.

It is useful to note here that the objective function in each instance incorporates a single decision variable to represent prediction error, in units of normalised RMS error, one decision variable to represent each potential input variable associated with each application and a single decision variable to represent computation time, in units of hours. Further, the coefficients of the decision variables are established to represent the cost associated with the appropriate units of the decision variables in each instance. Nevertheless, problem formulation for each of the studied industrial applications is complete. Hence, it is now appropriate to apply the developed optimisation technique to establish an optimal solution for each of the applied models in each instance and moreover, determine the most economically beneficial model to use for each application.

7.4 APPLICATION OF DEVELOPED OPTIMISATION TECHNIQUE

A systematic procedure is used to determine an optimal solution for each of the neural networks applied to each of the studied industrial applications. In particular, the procedure involves:

i). specification of appropriate predictive importance analysis results
ii). specification of relevant decision variable costs, and
iii). optimisation analysis to determine an optimal solution

i). Specification of appropriate predictive importance analysis results - The predictive importance analysis results associated with the neural network being optimised are required to be entered appropriately to a particular worksheet in the *Neural Network Analysis and Optimisation Strategy* program for reference during the optimisation analysis. While the procedure used to complete a predictive importance analysis is

documented in Chapter Three, the procedure used to enter the predictive importance analysis results is discussed in detail in Appendix D.

ii). Specification of relevant decision variable costs - In addition to the predictive importance analysis results, the relevant decision variable costs are required to be entered to a particular worksheet in the *Neural Network Analysis and Optimisation Strategy* program for reference during the optimisation analysis. The procedure used to enter the decision variable costs is discussed in detail in Appendix D.

iii). Optimisation analysis to determine optimal solution - Providing the predictive importance analysis results and relevant decision variable costs have been entered correctly to the appropriate worksheets, the optimisation analysis can be initiated and completed. Likewise, the procedure to achieve this is discussed in detail in Appendix D. It is useful to note that the optimisation analysis results show the original solution and associated cost and the optimal solution and associated cost. The cost per prediction noted for the original and optimal solutions are the corresponding values of the objective function following substitution of the decision variable values that are specified in the original and optimal solutions.

The procedure documented here is used to determine an optimal solution for each neural network applied to each of the studied industrial applications. In addition, it is also useful to note that selection of the optimal solution that yields the minimum value of the objective function for each application can be completed using the developed computer program. The procedure to achieve this is documented in Appendix D, while the optimal solution established for each neural network for each of the studied industrial applications using the developed optimisation technique is documented in the following section.

7.4.1 Optimisation Analysis Results for Electrolyte Additive Prediction Application

The original and optimal solutions for each of the neural network models applied to the electrolyte additive prediction application are documented in Appendix F, Tables F.1 to F.6. Substituting the values of the decision variables given in the original and optimal solutions established for each of the neural networks applied to the

electrolyte additive prediction application yields the value of the objective function shown in Equation 7.3.5. The corresponding values of this objective function for the original and optimal solutions for each applied model are documented in Table 7.4.1, with the difference between the original and optimal solution costs, denoted as saving, highlighted in each instance.

TABLE 7.4.1. Value of Objective Function for Original and Optimal Solutions for Neural Networks Applied to Electrolyte Additive Prediction Application

	Neural Network Model					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
z - original solution ($)	26.36	20.27	25.88	24.55	18.78	13.01
z - optimal solution ($)	18.66	15.16	17.10	14.38	13.88	11.82
saving ($)	7.70	5.11	8.78	10.17	4.90	1.19

Considering, firstly, the WH neural network, it can be seen that the optimal solution is an improvement on the original solution. Specifically, while the original solution yields a corresponding z value of $26.36 it is shown that the optimal solution yields an associated z value of $18.66, representing a saving of $7.70 per prediction. It is shown that this saving is achieved by removing bath temperature, AlF_3 and Na_2CO_3 addition (t-1) and Na content from the original solution. While there is a slight increase in RMS error as a result of removing these contributing process parameters from the WH neural network, it is shown that the decrease in process parameter measurement cost yields a substantial saving overall. Similarly, the BP1 network has shown some improvement in the value of the objective function following the optimisation analysis. It is shown that the original solution, yielding a z value of $20.27, has bath height and F content removed as a result of the optimisation analysis, resulting in a z value of $15.16. Consequently, the reduction in process parameter measurement cost plus the slight increase in associated RMS error cost yields a total saving of $5.11 per prediction. Moreover, the prediction cost associated with the BP1 network is shown to be lower than that associated with the WH neural network, using the original and optimal solution in either instance. However, while the BP2 network has shown a higher saving as a result of the optimisation analysis, the prediction cost associated with the BP2 network is higher than that associated with the BP1 network. In particular, it is shown that the original solution for the BP2 network has an associated z value of $25.88, while the optimal solution has an

associated z value of \$17.10, yielding a saving of \$8.78, attributed to the removal of Na_2CO_3 addition (t-1) and F and Na content from the original solution. On the other hand, the removal of bath height and F and Na content from the original solution of the RBF network yields an improved prediction cost that is lower than that associated with either the WH, BP1 or BP2 networks. Specifically, the original solution of the RBF network has an associated prediction cost of \$24.55, which is improved to obtain an optimal solution yielding an objective function value of \$14.38, resulting in a prediction cost saving of \$10.17. It is interesting to note that while the objective function value associated with the original solution for the RBF network was initially higher than that associated with the original solution for the BP1 network, \$24.55 and \$20.27, respectively, the optimisation analysis has resulted in the RBF network having an ultimately lower prediction cost than the BP1 network. This finding has major implications in regard to the economic benefit of neural network modelling. It is shown that while, in the first instance, the BP1 network has the lower associated prediction cost, completion of the optimisation analysis has shown that the RBF network can be manipulated to achieve a lower prediction cost. Further, while the BP1 network was also subjected to the optimisation analysis, the minimum prediction cost achievable using the BP1 network was higher than that associated with the optimal RBF model. Hence, this result highlights the importance and benefit of completing an optimisation analysis. It can be seen that the value of the objective function associated with the RBFKOH network is reduced as a result of the optimisation analysis. In particular, it is shown that the original solution for the RBFKOH network yields a z value of \$18.78, while the optimal solution yields a z value of \$13.88, resulting in a prediction cost saving of \$4.90, attributed to the removal of F content from the original solution. Similarly, the optimal solution established for the GRNN yields an objective function value of \$11.82, which is lower than that achieved using the original solution, shown to be \$13.01, attributed to the removal of bath temperature and AlF_3 addition (t-1) from the original solution. While this represents a prediction cost saving of only \$1.19, it is shown that the GRNN achieves the lowest objective function value of the neural networks applied to the electrolyte additive prediction application. It can be seen that the GRNN achieved the lowest original solution prediction cost and further, the lowest optimal solution prediction cost of the applied networks.

It is interesting to note that while the GRNN model achieved the lowest prediction cost of the applied neural networks it did not have the lowest prediction error. It is shown in Table 7.4.2 that the RBFKOH achieved the lowest prediction error of the applied networks. However, it is shown that the GRNN model requires only target bath temperature, bath resistivity and temperature reference to achieve the RMS error value of 0.0742, whereas the RBFKOH model requires target bath temperature, bath temperature, bath resistivity, AlF_3 addition (t-1), cell power, cell age and temperature reference to achieve the lower RMS error of 0.0723. Hence, the overall cost per prediction is lower using the GRNN model due to the lower process parameter measurement cost. In addition, it can be seen that while the RBF network required the same process parameters as the RBFKOH model, the RBF network had a higher RMS error and therefore a higher prediction cost. Similarly, although the WH network required the same process parameters as the GRNN model, the WH network had a higher associated RMS error value and therefore a higher prediction cost. It is also interesting to note that the BP1 model had lower prediction error and computation time than the BP2 network and further, required less process parameters to achieve the lower RMS error. Nevertheless, it can be seen that the removal of contributing process parameters from the original solution resulted in an increase in RMS error and decrease in computation time in the optimal solution in each instance. It has been noted that the magnitude of the RMS error increase in the optimal solution is a function of the particular parameters omitted from the original solution. On the other hand, computation time decrease is an average value and is not dependent on which parameters are removed from the original solution, but rather, how many parameters are removed.

TABLE 7.4.2. Summary of Optimal Solutions for Neural Networks Applied to Electrolyte Additive Prediction Application

Decision Variable	Neural Network Optimal Solution (Units)					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
x_1	0.1172	0.0804	0.0873	0.0755	0.0723	0.0742
x_2	1	1	1	1	1	1
x_3	0	1	1	1	1	0
x_4	0	0	1	0	0	0
x_5	1	1	1	1	1	1
x_6	0	0	1	0	0	0
x_7	0	0	1	1	1	0
x_8	0	1	0	0	0	0
x_9	0	1	1	1	1	0

x_{10}	0	1	1	1	1	0
x_{11}	0	0	0	0	0	0
x_{12}	0	0	0	0	0	0
x_{13}	1	1	1	1	1	1
x_{14}	0.06	0.24	0.77	0.31	0.73	0.01

While the value of the objective function for each neural network applied to the electrolyte additive prediction application has been established in each instance, it is also interesting to consider the multi-variable regression analysis (MVRA) model in this discussion. However, an optimal solution is not established for the MVRA as the objective of this particular work is to investigate neural network optimisation. However, it is useful to consider the MVRA model for comparison with the applied neural networks. Hence, the value of the objective function for the MVRA model is calculated for the RMS error, number of process parameters required and computation time established for the MVRA model in the first instance. This information is used to formulate the original solution for the MVRA model, as documented in Appendix F, Table F.7. It is shown that the RMS error associated with the MVRA model is 0.1129, while target bath temperature, bath temperature, bath resistivity, emf, AlF_3 addition (t-1) and cell power are contributing process parameters and the corresponding computation time is 5.00 seconds, rounded to 0.00 hours in the original solution. Substituting the appropriate decision variable values into the objective function given in Equation 7.3.5 yields a z value of $20.62. Observation of the MVRA solution shows that the total measurement cost of the process parameters that may be omitted from the original solution yield a total cost of $2.67. Hence, ignoring the increase in RMS error cost that would result from the removal of the contributing process parameters from the original solution, it is not possible to reduce the prediction cost of the MVRA model lower than $17.95. Consequently, the MVRA model is the least economically beneficial modelling paradigm for this particular application. It is shown that the optimal solution prediction cost associated with each of the neural networks is lower than $17.95, with the exception of the WH model. However, the MVRA RMS error would only need to increase by approximately 4.0% as a result of the removal of the contributing process parameters from the original solution to increase the associated prediction cost of the MVRA model above the optimal solution prediction cost of the WH network, which is probable.

7.4.2 Optimisation Analysis Results for Cell Failure Prediction Application

The developed optimisation technique is applied to each of the neural networks applied to the cell failure prediction application to obtain an optimal solution in each instance. Consequently, the original and optimal solutions for each neural network are documented in Appendix F, Tables F.8 to F.13 for the cell failure prediction application. In addition, the values of the decision variables specified in the original and optimal solutions are substituted into the objective function given in Equation 7.3.6 to calculate the z values documented in Table 7.4.3.

TABLE 7.4.3. Value of Objective Function for Original and Optimal Solutions for Neural Networks Applied to Cell Failure Prediction Application

	Neural Network Model					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
z - original solution ($)	62.50	50.48	57.26	87.32	69.33	102.88
z - optimal solution ($)	43.30	50.48	51.26	87.32	68.74	83.50
saving ($)	19.20	0.00	6.00	0.00	0.59	19.38

For the WH network, it is shown that the value of the objective function as a result of the optimisation analysis is significantly lower than that associated with the original solution. It is shown that the z value of $62.50 associated with the original solution is reduced to $43.30 when an optimal solution is established. While there is a slight increase in error cost associated with the optimal solution, the removal of Fe and Si content (weekly), bath temperature, rod height, Si and Fe content, Fe/V and low temperature count from the original solution yield the noted prediction cost saving of $19.20. Moreover, it can be seen that the objective function value associated with the optimal solution for the WH network is the lowest of all the applied neural networks. It is especially valuable to note that while the WH network did not have the lowest original solution prediction cost, the optimisation analysis has established a unique combination of RMS error, process parameters required and computation time to give the WH network the lowest prediction cost. On the other hand, the BP1 network has shown no reduction in prediction cost as a result of the optimisation analysis. It is shown that the original and optimal solutions both yield an objective function cost of $50.48. This result implies that the removal of any process parameters from the original solution yields an increased error cost that is greater than the measurement cost of the process parameter in each instance. Hence, it is shown that it is not

economically beneficial to remove any process parameters from the BP1 network. While the BP1 network has shown the lowest original solution prediction cost, it was not possible to improve on the value of the BP1 network objective function value to achieve a lower prediction cost than the WH network. The BP2 network has a *z* value of $57.26 associated with the original solution, which is shown to decrease to $51.26 for the optimal solution, yielding a prediction cost saving of $6.00. This saving is attributed to the exclusion of bath temperature, Si and Fe content, Fe/V and Fe/Ga from the original solution. Similar to the BP1 network, the RBF network has shown no improvement in prediction cost as a result of the optimisation analysis. It is shown that the prediction cost associated with the original and optimal solution in each instance is $87.32. Consequently, the removal of any process parameters from the original solution does not yield a decrease in prediction cost. However, the value of the objective function associated with the original solution for the RBFKOH network, shown to be $69.33, is slightly lower for the optimal solution, shown to be $68.74, representing a prediction cost saving of $0.59. The abstraction of Fe/V from the original solution is responsible for the prediction cost saving. Considering the application of the GRNN to the cell failure prediction application, it is shown that the prediction cost saving as a result of the optimisation analysis has the highest magnitude of all the applied neural networks. In particular, a prediction cost saving of $19.38 is the result of a reduction in *z* value from $102.88 associated with the original solution to $83.50 for the optimal solution. It can be seen that the prediction cost saving is attributed to the exclusion of Fe and Si content (weekly), bath height, Si and Fe content, Fe/V and high and low temperature count from the original solution. It is interesting to note that while the GRNN has shown the highest prediction cost saving as a result of the optimisation analysis, the GRNN does not achieve the lowest optimal solution prediction cost of the applied neural networks, attributed to the high original solution prediction cost.

It is also interesting to note that while the WH network has shown the lowest prediction cost of the applied models, the BP1 model in fact achieved the lowest RMS error, as shown in Table 7.4.4. It is shown, however, that the WH network requires substantially less process parameters than the BP1 network, therefore reducing prediction cost. Similarly, the BP2 network has also shown lower RMS error than the WH network, however, the BP2 model has a higher associated process

parameter measurement cost due to the high number of process parameters used in the optimal solution. Further, it is shown that the RBF and RBFKOH models have a comparatively high associated prediction cost due to the high prediction error and high number of process parameters used in the optimal solution. On the other hand, while the GRNN model uses only four process parameters in the optimal solution, the high prediction error yields a comparatively high associated prediction cost. It is also interesting to note for the WH, BP2, RBFKOH and GRNN models that the removal of particular process parameters from the original solution yields an increase in RMS error in the optimal solution and a slight decrease in computation time in each instance. This is expected behaviour as the optimisation analysis incorporates specific procedures to accurately simulate model behaviour as a result of the removal of process parameters from the neural network.

TABLE 7.4.4. Summary of Optimal Solutions for Neural Networks Applied to Cell Failure Prediction Application

Decision Variable	Neural Network Optimal Solution (Units)					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
x_1	0.0818	0.0451	0.0789	0.1279	0.1139	0.1650
x_2	0	1	1	1	1	0
x_3	0	1	1	1	0	0
x_4	1	1	1	1	1	1
x_5	0	1	0	0	0	0
x_6	0	1	0	1	1	0
x_7	1	1	1	1	1	0
x_8	1	1	1	1	1	0
x_9	1	1	1	1	0	0
x_{10}	1	1	1	1	1	0
x_{11}	0	1	1	1	1	0
x_{12}	1	1	1	1	1	0
x_{13}	1	1	1	1	1	1
x_{14}	0	1	1	1	1	1
x_{15}	1	1	1	1	0	0
x_{16}	0	1	0	0	0	0
x_{17}	1	1	1	0	0	0
x_{18}	1	1	1	0	0	1
x_{19}	1	1	1	1	1	0
x_{20}	0	1	0	0	0	0
x_{21}	0	1	0	1	1	0
x_{22}	0	1	0	1	0	0
x_{23}	0	1	0	1	0	0
x_{24}	1	1	1	1	1	0
x_{25}	0	0	1	0	0	0
x_{26}	0.02	1.23	0.70	0.50	1.24	0.01

The original solution for the MVRA model for this application is documented in Appendix F, Table F.14. While an optimisation analysis has not been completed for the MVRA model, the original solution is established in this instance from the combination of RMS error, number of process parameters used and computation time associated with the initial MVRA modelling completed for this application. Substituting the values of the MVRA original solution into the objective function given in Equation 7.3.6 yields a prediction cost of $177.34, which is significantly higher than the prediction cost associated with each of the applied neural network models. Moreover, removal of the contributing process parameters from the original solution for the MVRA model would result in a prediction cost of $163.75, not considering the RMS error increase that would result as a consequence. Hence, it follows that it is not possible to reduce the prediction cost associated with the MVRA model below a value that is substantially higher than the prediction cost associated with each of the neural networks applied to the cell failure prediction application. Hence, similar to the previous application, the MVRA is the least preferred modelling technique in this instance as it yields minimum economic benefit.

7.4.3 Optimisation Analysis Results for Electrolyte Temperature Prediction Application

Each of the neural networks applied to the electrolyte temperature prediction application are subjected to the optimisation analysis to ascertain the optimum neural network prediction cost in each instance. The original and optimal solutions established for each applied network are documented in Appendix F, Tables F.15 to F.20. Substituting the values of the decision variables given in the original and optimal solutions into the objective function specified in Equation 7.3.7 yields the z values and the associated prediction cost savings documented in Table 7.4.5.

TABLE 7.4.5. Value of Objective Function for Original and Optimal Solutions for Neural Networks Applied to Electrolyte Temperature Prediction Application

	Neural Network Model					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
z - original solution ($)	48.50	45.54	45.04	48.88	47.44	51.84
z - optimal solution ($)	48.04	45.54	45.04	48.61	47.44	51.84
saving ($)	0.46	0.00	0.00	0.27	0.00	0.00

While the original solution yields an associated objective function value of \$48.50 it is shown that a slight improvement in z is achieved using the optimal solution, shown to be \$48.04. The prediction cost saving of \$0.46 is shown to be attributed to the removal of AlF_3 addition from the original solution. Conversely, applying the developed optimisation technique to the original solution established for the BP1 and BP2 neural networks has shown no improvement in the value of the objective function. It is shown that the z value of \$45.54 associated with the original solution for the BP1 network does not change as a result of the optimisation analysis. Likewise, the BP2 network has shown no improvement from the original solution objective function value of \$45.54 as a result of the optimisation analysis. Hence, the optimisation analysis has shown for the BP1 and BP2 networks that it is not economically feasible to exclude any process parameters from the original solution. However, it is interesting to note that the BP2 network has shown the lowest prediction cost of the neural networks applied to the electrolyte temperature prediction application. It is shown that the optimal solution prediction cost of \$45.04 is lower than that achieved by any other neural network. While the BP2 network has shown no improvement in prediction cost as a result of the optimisation analysis, the low prediction cost associated with the BP2 network is attributed to the low z value corresponding to the original solution. Similar to the WH network, the RBF network has shown some improvement in the value of the objective function as a result of the optimisation analysis. It is shown that the original solution prediction cost of \$48.88 is reduced to an optimal solution prediction cost of \$48.61, yielding a prediction cost saving of \$0.27, attributed to the exclusion of bath height from the original solution. On the other hand, applying the developed optimisation technique to the RBFKOH and GRNN models for this application has shown that it is not economically beneficial to remove any process parameters from the original solution in either instance. Specifically, for the RBFKOH network, the objective function value associated with the original solution, shown to be \$47.44 does not decrease as a result of the optimisation analysis. That is, the optimal solution prediction cost is also shown to be \$47.44. Hence, it is not possible to achieve a prediction cost reduction by removing process parameters from the original solution of the RBFKOH network. Similarly, the value of the objective function for the original solution for the GRNN, shown to be \$51.84, is the same as the z value for the GRNN optimal solution. Hence, removal of process parameters from the original solution obtained for the

GRNN does not yield a lower prediction cost. It is also interesting to note that the GRNN has the highest associated prediction cost of the neural networks applied to the electrolyte temperature prediction application. This is attributed to the fact that the GRNN model has the highest prediction error of the applied models and further, has a high number of process parameters in the optimal solution, as shown in Table 7.4.6. On the other hand, it can be seen that while the BP2 network requires all of the potential process parameters as network inputs, the low prediction error associated with this model results in the BP2 network having the lowest prediction cost of the applied neural networks. It is interesting to note that due to the high value of the coefficient c_1 in the objective function given in Equation 7.3.7, prediction error plays a significant role in determining the prediction cost associated with each model. In particular, it can be seen that the BP1 network has a higher RMS error than the BP2 network and therefore a higher prediction cost, as do the WH, RBF and RBFKOH neural networks. While the optimisation analysis has resulted in an improved original solution for the WH and RBF neural networks, it is shown that the removal of particular process parameters from the original solution in either instance yields a slight increase in RMS error and decrease in computation time. However, it is shown that the cost associated with the resulting increase in RMS error and decrease in computation time is lower than the cost associated with the removal of specific process parameters, resulting in a slightly improved original solution in either instance.

TABLE 7.4.6. Summary of Optimal Solutions for Neural Networks Applied to Electrolyte Temperature Prediction Application

Decision Variable	Neural Network Optimal Solution (Units)					
	WH	BP1	BP2	RBF	RBFKOH	GRNN
x_1	0.0636	0.0625	0.0573	0.0669	0.0640	0.0702
x_2	1	1	1	1	1	1
x_3	1	1	1	1	1	1
x_4	0	0	1	0	1	1
x_5	1	0	1	1	0	1
x_6	1	0	1	0	1	1
x_7	1	1	1	1	1	1
x_8	1	1	1	0	0	1
x_9	1	1	1	0	1	1
x_{10}	0	0	1	0	0	0
x_{11}	1	0	1	0	0	0
x_{12}	0.04	0.18	0.69	0.25	0.87	0.01

It is shown in Appendix F, Table F.21 that the prediction cost associated with the MVRA is \$53.86. This figure is calculated by substituting the values of the decision variables specified in the original solution into the objective function given in Equation 7.3.7. It is shown that the RMS error associated with the MVRA model is higher than that associated with any of the neural networks applied to this particular application. Further, it can be seen that the MVRA incorporates all of the process parameters, with the exception of Na_2CO_3 addition, in the optimal solution. Consequently, the value of the objective function is higher for the MVRA than it is for the applied neural networks. In addition, due to the increase in error that would result from the removal of contributing process parameters from the original solution of the MVRA model, it is likely that the prediction cost associated with the MVRA model would never be lower than that associated with each of the neural networks. Specifically, consider the prediction cost associated with the MVRA model with only cell age as an input variable, which is shown to have a measurement cost of \$0.00. The prediction cost would be \$50.62 without considering increased error cost as a consequence. However, it would only require an error increase of approximately 2.5%, which is highly likely, to increase the MVRA prediction cost above the highest neural network prediction cost of \$51.84, associated with the GRNN. Hence, it is estimated that the prediction cost associated with the MVRA model will be higher than that associated with the neural networks applied for electrolyte temperature prediction and does not have the potential to ever be lower, using the specified objective function. Hence, the MVRA would not be a suitable preference as a modelling paradigm for the electrolyte temperature prediction application.

7.5 CONFIRMATION OF OPTIMAL SOLUTIONS FOR NEURAL NETWORKS APPLIED TO STUDIED INDUSTRIAL APPLICATIONS

It has been noted that the developed optimisation technique incorporates specific routines to ensure that the removal of contributing process parameters from the original solution results in an appropriate and accurate increase and decrease in RMS error and computation time, respectively. However, it has also been noted that while these routines are incorporated into the optimisation analysis it is necessary to confirm that an optimal solution is achievable by re-training the neural network for which the optimal solution is established. This is completed by training and testing the neural network using only those process parameters that are assigned a value of 1

in the optimal solution, as these are the parameters that are nominated for inclusion in the optimum neural network model. Consequently, the RMS error and computation time associated with the re-trained neural network model should be comparable to that specified in the optimal solution. While potential consequences of re-training the neural network have been identified, the procedures to overcome such consequences have also been discussed.

A particular procedure is applied to confirm the capability of a neural network to achieve the established optimal solution. This procedure is written as follows:

Step 1. Create train and test data sets that include only those process parameters assigned a value of 1 in the optimal solution.

Step 2. Complete sufficient training iterations to allow network convergence using the optimum neural network architecture identified prior to the predictive importance analysis.

Step 3. Identify the minimum test error and computation time associated with the re-trained neural network.

Step 4. Compare the minimum test error and computation time associated with the re-trained neural network with that established in the optimal solution.

Step 1. Create train and test data sets - The train and test data sets used for re-training the neural network model are a modification of the original train and test data sets used for initial neural network training. Specifically, the train and test data sets developed for re-training have the process parameters assigned a value of 0 in the optimal solution omitted. Hence, while the dimension of the original train and test data sets is i, where i = number of process parameters used as potential neural network input variables, the modified train and test data sets for re-training have dimension i', where i' = number of process parameters assigned a value of 1 in the optimal solution. It is important to note that the train and test data sets used for neural network re-training have the same number of train and test patterns used for initial training and testing. Hence, the train and test data sets used for initial neural network training and re-training differ only by the number of process parameters included in the data sets.

Step 2. Complete sufficient training iterations - The number of training iterations completed by a neural network must be sufficient to allow convergence of the network weights. It has been shown that neural network prediction error decreases with an increasing number of iterations and reaches a minimum value when network weight convergence occurs. Hence, in order to achieve minimum prediction error with a neural network it is critical that sufficient iterations be completed to allow network weight convergence. This is achieved by observing error behaviour with an increasing number of iterations to identify the point where error becomes uniform and noting the corresponding number of iterations. The architecture and algorithm used for re-training each neural network are the same as the minimum error yielding conditions determined prior to the predictive importance analysis. However, the number of input nodes in each model is equivalent to the number of parameters assigned a value of 1 in the optimal solution.

Step 3. Identify the minimum test error and computation time - Observation of RMS error with an increasing number of iterations identifies the minimum error achieved by a neural network. Further, the computation time corresponding to the iteration where the minium error occurred can be established.

Step 4. Compare the minimum test error and computation time - The minium test error and computation time associated with the re-trained neural network is compared with the error and computation time specified in the optimal solution. However, a statistical test is not appropriate in this instance for comparison of the error specified in the optimal solution with the error associated with the re-trained neural network model. The error specified in the optimal solution is a single value and does not constitute a valid population size for statistical comparison.

It is useful to note in instances where the optimal solution is equivalent to the original solution there is no requirement to re-train the neural network. This is due to the fact that the optimal solution is already confirmed as being achievable by the corresponding neural network because it is based on the findings of the associated predictive importance analysis. Nevertheless, the procedure outlined here is applied where necessary to re-train the neural networks for each of the studied industrial applications.

7.5.1 Confirmation of Optimal Solutions for Electrolyte Additive Prediction Application

For the electrolyte additive prediction application it is shown that the established optimal solution for each applied neural network is an improvement on the original solution. Hence, it is necessary to re-train each of the applied neural networks to confirm their capability to achieve the established optimal solution. A comparison of the optimal solution and re-trained neural network in each instance is shown in Appendix G, Tables G.1 to G.6. It is necessary to note for this particular application that the train and test sets for each neural network consist of 1,365 and 200 data patterns, respectively. However, the particular process parameters used in the data set for each neural network correspond to the particular process parameters nominated in the optimal solution for each neural network. Further, it has been noted that the neural network architecture and algorithm used for re-training is the optimum architecture identified prior to the predictive importance analysis. Specifically, for the WH neural network it is shown that the sigmoidal activation function in the output layer of the network produced lower error than the linear summation function. Hence, the sigmoidal activation function is used in this instance. It is shown that the re-trained WH model achieves a minimum test error of 0.1170, which is comparable to the test error of 0.1172 nominated by the optimal solution. Further, the computation time associated with the re-trained WH model is the same as that given in the optimal solution. Similarly, the re-trained BP1 and BP2 models yield a minimum RMS error of 0.0807 and 0.0872, respectively, which is similar to that specified by the optimal solution in each instance. It is useful to note that the BP1 network incorporated 8 hidden layer nodes, while the sigmoidal activation function was used in the hidden and output layers of the network. While the sigmoidal activation function was also used in the hidden and output layers of the BP2 model, the optimum architecture has been shown to be 8 nodes in the first hidden layer and 6 nodes in the second hidden layer of the network. The re-trained RBF and RBFKOH neural networks have also shown comparable error and computation time to that given in the optimal solution. While the RBF and RBFKOH models were both re-trained using the Gaussian function in the hidden layers of the network, both networks were re-trained using the previously determined optimum architecture of 20 hidden layer nodes and receptive field width of 0.9. Further, the linear summation function was used in the output layer nodes in either instance. The GRNN has shown

a test RMS error of 0.0743 after re-training using only those process parameters nominated in the optimal solution, which is similar to the error value of 0.0742 calculated during the optimisation analysis. The computation time associated with the re-trained GRNN is also comparable to that given in the optimal solution, which is attributed to the fact that the GRNN model requires no iterations during training, hence, there is minimum opportunity for discrepancy. Re-training for the GRNN was completed in this instance using 600 pattern layer nodes incorporating the exponential activation function and a receptive field width of 0.1.

It is interesting to note for each of the re-trained neural networks that network convergence was achieved in approximately the same number of iterations as completed in the initial training phase. Hence, there was no discrepancy between computation time specified for a neural network in the optimal solution and computation time associated with re-training the corresponding neural network. This finding has major implications in regard to the developed optimisation technique. An assumption is made in the optimisation analysis that the same number of iterations required in the initial training phase is sufficient for neural network re-training. Specifically, the optimisation analysis, when calculating the computation time for the optimal solution, does not consider the possibility that a longer training period may be required. While the assumption was based on numerous investigations completed prior to development of the optimisation technique, the results presented here have confirmed the accuracy of the assumption.

7.5.2 Confirmation of Optimal Solutions for Cell Failure Prediction Application

It is shown that the optimisation analysis has resulted in an improved solution for the WH, BP2, RBFKOH and GRNN models for the cell failure prediction application. A comparison of the optimal solution and re-trained neural network in each instance is shown in Appendix G, Tables G.7 to G.12. It is useful to note that while there is no requirement to re-train the BP1 and RBF networks in this instance, the optimal solution and neural network units are presented for these models to provide continuity and consistency in the documentation. While the dimension of the train and test data sets for re-training is determined for each neural network by the optimal solution, the number of train and test data patterns in each instance was 1,500 and 500, respectively. It is shown that the re-trained WH model, using the sigmoidal

activation function in the output layer of the network, achieved a test RMS error of 0.0815 and computation time of 0.02 hours, which are comparable to the error and computation time calculated in the optimal solution. Likewise, the re-trained BP2 network has shown no disagreement with the error and computation time associated with the optimal solution. It is shown that the BP2 neural network achieved a RMS error of 0.0787 after re-training, which is comparable to the error value of 0.0789 calculated during the optimisation analysis. Further, the computation time required in either instance is also shown to be approximately similar. The BP2 neural network was re-trained using 6 and 5 nodes in the first and second hidden layers, respectively while the sigmoidal activation function was used in all processing nodes. The RBFKOH model has been confirmed as capable of achieving a minimum test RMS error and computation time that is similar to that given in the optimal solution. The RBFKOH, re-trained using the Gaussian activation function in the 40 hidden layer nodes and the linear summation function in the output layer node has shown a RMS error value of 0.1142 after re-training, which is similar to the optimal solution RMS error value of 0.1139. In addition, computation time is shown to be equal to 1.24 hours in either instance. The RMS error value of 0.1648 associated with the re-trained GRNN model, using 400 pattern layer nodes incorporating the exponential function and a receptive field width of 0.1, is also shown to be comparable to the RMS error value of 0.1650 specified in the optimal solution. Similar to the previous application, the re-trained neural networks have shown similar computation time in each instance to that specified in the optimal solution. This is attributed to the fact that the required number of iterations for network convergence was comparable to that required in the initial training phase in each instance.

7.5.3 Confirmation of Optimal Solutions for Electrolyte Temperature Prediction Application

It has been shown for the electrolyte temperature prediction application that the WH and RBF networks were the only models of the applied neural networks to achieve an optimal solution that was an improvement on the original solution. However, similar to the previous application, for continuity and consistency in the documentation the optimal solution and neural network units are acknowledged for each of the applied models, as shown in Appendix G, Tables G.13 to G.18. The train and test data sets consisted of 1,244 and 200 data patterns, respectively. It is shown for the WH

network that the re-trained model achieved a test RMS error of 0.0639, which is similar to the RMS error value of 0.0636 specified in the optimal solution. In addition, computation time is shown to be comparable in either instance. It is useful to note that the sigmoidal activation function was used in the output layer of the WH model as this particular activation function has been shown to yield the lowest error in the WH network of the studied activation functions. Similarly, the re-trained RBF network has shown no disagreement with the RMS error calculated in the optimisation analysis and further, computation time given in the optimal solution is approximately the same for the re-trained model. The optimum architecture identified for the RBF model and therefore used for re-training consisted of 30 hidden layer nodes and a receptive field width of 0.3. Further, the Gaussian activation function and linear summation function were used in the hidden and output layer nodes, respectively. It is noted that it is not necessary to re-train the remaining neural networks as the optimal solution has been confirmed as achievable as a result of the predictive importance analysis completed in each instance. Specifically, the optimal solution established for the BP1, BP2, RBFKOH and GRNN models is equivalent in each instance to the combination of RMS error, number of process parameters and computation time identified for each neural network as a result of the predictive importance analysis. Consequently, re-training for confirmation of the optimal solution is not necessary for these models. Nevertheless, it is interesting to note that both the WH and RBF neural networks achieved network convergence in approximately the same number of iterations during re-training as that required in the initial training phase. This again confirms the accuracy of the assumption made in the developed optimisation technique that network convergence is achieved in approximately the same number of iterations in either instance.

7.5.4 Summary of Confirmation of Optimal Solutions for Applied Neural Networks

It is shown for each of the neural networks applied to each of the studied industrial applications that the optimal solution is achievable in each instance. For each of the neural network models, for which the optimal solution is an improvement on the original solution, verification of the optimal solution is required. It is noted that this is completed by re-training the corresponding neural network in each instance to achieve a comparable combination of RMS error, number of process parameters and

computation time as that given in the corresponding optimal solution. Further, for those neural networks applied to the cell failure and electrolyte temperature prediction applications that have shown no improvement in the original solution as a result of the optimisation analysis, it is noted that confirmation of the optimal solution is not required. In these instances, the optimal solution is equivalent to the original solution, which is based on the findings of the predictive importance analysis and therefore confirmed as achievable.

Hence, as it is verified that each of the optimal solutions obtained for each of the neural networks applied to the studied industrial applications is achievable using the corresponding neural network, it is now appropriate to select a particular neural network to implement for each application. This selection process is based purely on a study of the prediction cost associated with each model. Further, it is useful to note that the regression model is included as a potential model for implementation in this economic analysis. The results of this selection process are outlined in the following section.

7.6 SELECTION OF OPTIMUM MODEL FOR STUDIED INDUSTRIAL APPLICATIONS

In order to select a particular neural network model for each of the studied industrial applications it is now useful to rank the applied models in order of increasing prediction cost, as shown in Table 7.6.1. It is important to note that the prediction cost documented in each instance is the value of the objective function for each application corresponding to the optimal solution identified for each model. Hence, the prediction cost associated with each applied model is obtained in each instance by substituting the decision variable values identified in the optimal solution into the corresponding objective function developed for each application. It is useful to note that this calculation is completed automatically using the developed optimisation technique. For the MVRA model, however, it is noted that the prediction cost documented in the following table is the value of the objective function calculated by substituting the values of the decision variables from the initial feasible solution into the appropriate objective function developed for each application. It has been noted that it is beyond the scope of this work to identify an optimal solution for the MVRA model. Nevertheless, it can be seen for each application that the applied models are

ranked from first to seventh, whereby the model with the lowest associated prediction cost is ranked first while the model exhibiting highest prediction cost is ranked seventh. Hence, the model with the lowest associated prediction cost is highlighted as the most appropriate model to use for the corresponding application while the model with the highest associated prediction cost is ranked as the least appropriate model to apply for the corresponding application. A discussion of the model ranking is completed for each of the studied industrial applications in the following section.

TABLE 7.6.1. Ranking of Models Applied to Studied Industrial Applications Based on Prediction Cost

	Electrolyte additive prediction application		Cell failure prediction application		Electrolyte temperature prediction application	
Model Ranking	Model	Cost ($)	Model	Cost ($)	Model	Cost ($)
1st	GRNN	11.82	WH	43.30	BP2	45.04
2nd	RBFKOH	13.88	BP1	50.48	BP1	45.54
3rd	RBF	14.38	BP2	51.26	RBFKOH	47.44
4th	BP1	15.16	RBFKOH	68.74	WH	48.04
5th	BP2	17.10	GRNN	83.50	RBF	48.61
6th	WH	18.66	RBF	87.32	GRNN	51.84
7th	MVRA	20.62	MVRA	177.34	MVRA	53.86

For the electrolyte additive prediction application it is shown that the GRNN model has the lowest associated prediction cost and is therefore ranked as most suitable for the application. It is interesting to note that the RBF and RBFKOH neural networks, shown to be similar models, have shown similar ranking for the application, second and third. Similarly, the BP1 and BP2 networks, which incorporate the same training algorithm and differ only by the number of hidden layers used in the network architecture, have also shown similar ranking, fourth and fifth. In this instance, the WH neural network is ranked sixth for the application, while the MVRA model is ranked as least suitable. Hence, it is shown that the optimum model to use for the electrolyte additive prediction application is the GRNN model. Further, it has been established that the necessary process parameters to use for electrolyte additive prediction are target bath temperature, bath resistivity and temperature reference, while 600 pattern layer nodes are required, as shown in Figure 7.6.1.

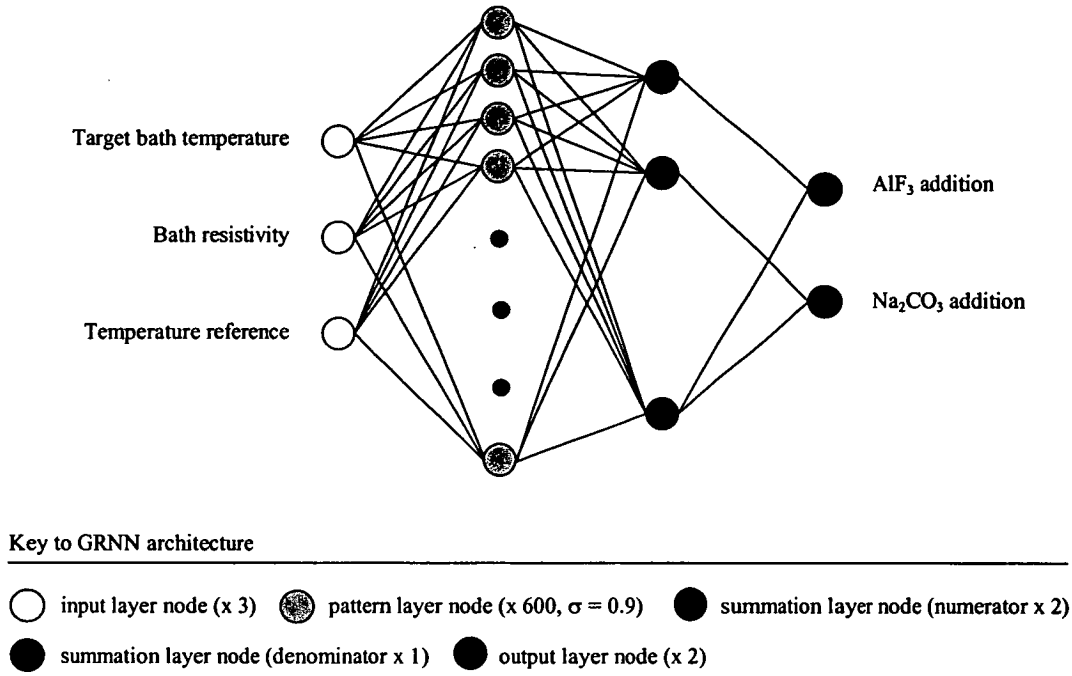


Fig. 7.6.1. Optimum Neural Network Model (GRNN) for Electrolyte Additive Prediction Application

Of the models applied to the cell failure prediction application, it is shown in Table 7.6.1 that the WH neural network has been identified as the optimum model to use. It is shown that the unique combination of prediction error, number of process parameters required and computation time associated with the WH model yields minimum prediction cost. Specifically, it is shown in Figure 7.6.2 that the process parameters used in the WH model are lining voltage drop, cell age, AE frequency and duration, unscheduled anode change, cell power, high frequency noise, cell resistance, bath resistivity, cell voltage, AE energy and high temperature count. Further, it has been noted that the sigmoidal function used in the output layer of the network yields minimum prediction error of the studied activation functions. It is interesting to note that the WH model had a significantly lower prediction cost than the second ranked model, the BP1 network, which has shown similar prediction cost to the BP2 model. Further, it is shown that the RBFKOH was ranked fourth for the application, followed by the GRNN and RBF models. Similar to the previous application, the MVRA model is ranked as least suitable for the application, in terms of economic benefit. Moreover, the MVRA model in this instance has exhibited a significantly higher prediction cost than the applied neural networks.

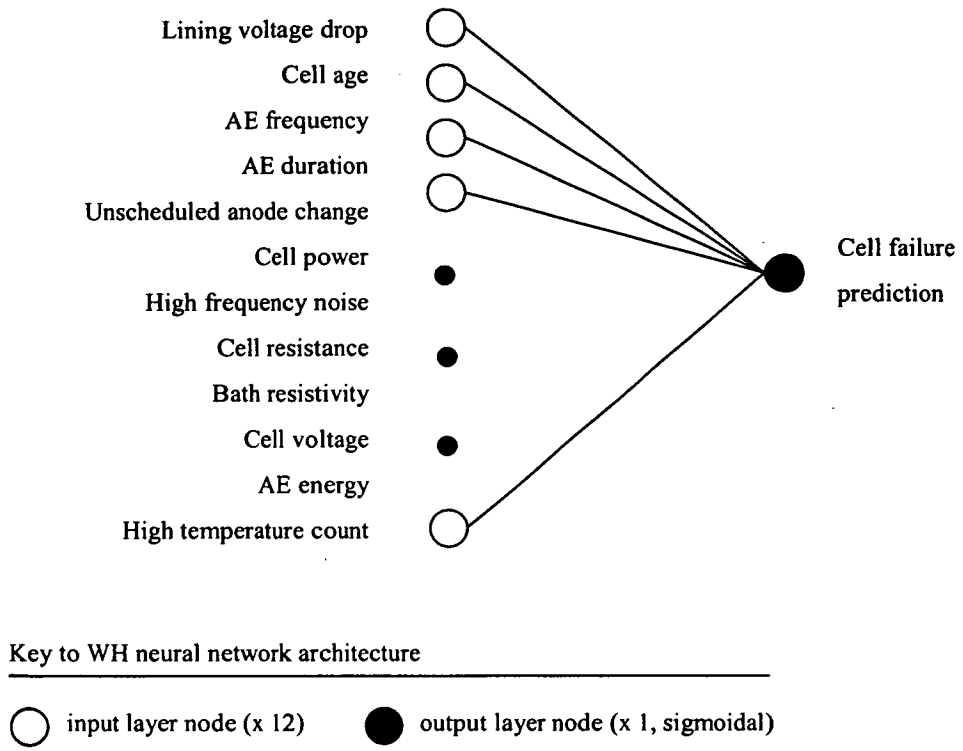


Fig. 7.6.2. Optimum Neural Network Model (WH) for Cell Failure Prediction Application

Considering the electrolyte temperature prediction application, it is shown in Table 7.6.1 that the BP1 and BP2 neural networks have shown similar prediction cost. However, for the electrolyte temperature prediction application it is shown that the prediction cost associated with the BP2 network is slightly lower than that associated with the BP1 model. While the BP2 neural network uses all of the potential process parameters as input variables in the developed model for electrolyte temperature prediction, it is shown that the BP2 model has the lowest associated prediction cost of the applied models. It is shown that electrolyte temperature (t-1), bath resistivity, electromotive force, cell age, bath height, high frequency noise, low frequency noise, cell power, AlF_3 addition and Na_2CO_3 addition are required input variables in the BP2 model, as shown in Figure 7.6.3. Moreover, the optimum architecture for the BP2 network incorporates 10 nodes in the first hidden layer and 3 nodes in the second hidden layer. In addition, all nodes used in the hidden and output layers of the BP2 model use the sigmoidal activation function. The RBFKOH has shown a slightly higher prediction cost than the BP1 neural network and is therefore ranked third for the application. Further, the WH and RBF networks are ranked fourth and fifth respectively, while the GRNN model is ranked last of the applied neural networks.

The MVRA model has again shown the highest associated prediction cost of the applied models and is therefore ranked as least suitable for the application.

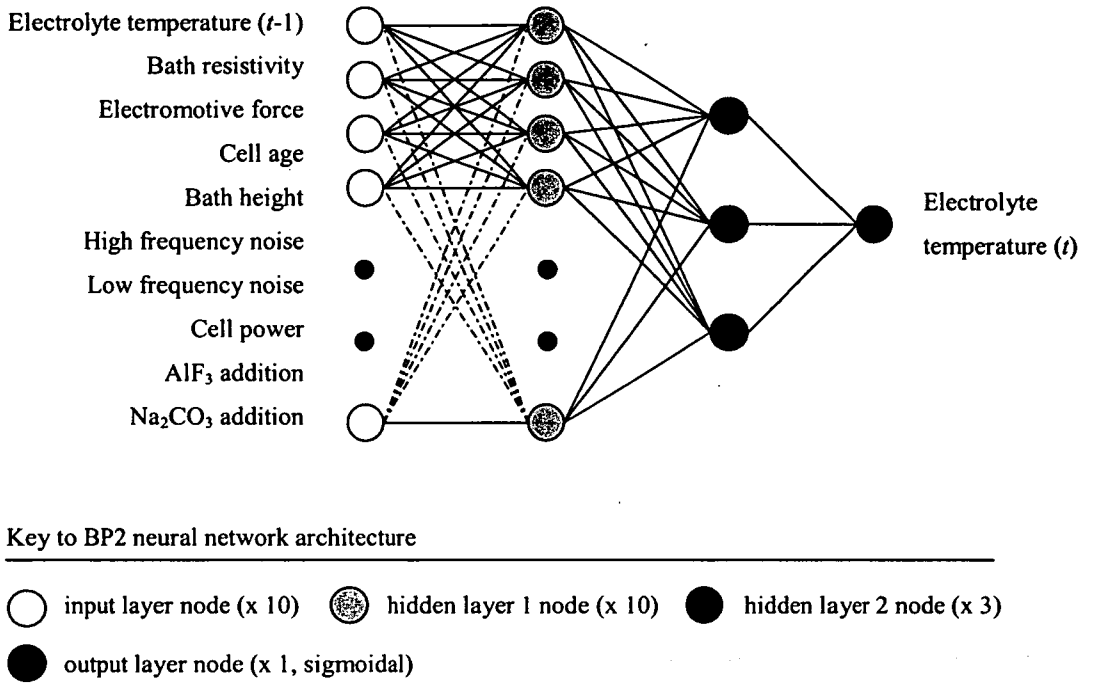


Fig. 7.6.3. Optimum Neural Network Model (BP2) for Electrolyte Temperature Prediction Application

7.7 CONCLUDING REMARKS

The problem considered in this chapter involves optimising the numerical value of a function of many variables, which is subject to particular constraints. However, the constraints are not formally specified but rather inferred from an associated and previously completed analysis. The importance of completing the optimisation analysis is highlighted by considering the results documented in this chapter. It is shown that the most economically beneficial model to use for an application is not necessarily the model that achieves minimum prediction error. Rather the most suitable model is that which achieves an optimum combination of error, process parameters used and computation time to yield minimum prediction cost. Hence, in the aluminium smelting industry, where economic benefit is a critical consideration, it is not appropriate to select a process model based purely on a study of prediction error.

It is useful to note here that the work completed has resulted in the development of a systematic procedure to select an optimum neural network model for an application. The procedure outlined here ensures that the most economically beneficial model is implemented for a particular application. Specifically, the procedure incorporates the following:

Step 1. Determine an optimum architecture for each applied neural network by observing RMS error behaviour with changing architecture and algorithm and selecting that architecture and algorithm that yield minimum RMS error.

Step 2. Complete a predictive importance analysis for each neural network using the optimum architecture and algorithm.

Step 3. Develop a unique objective function for the application that represents the selection criteria algebraically.

Step 4. Apply the developed optimisation technique to minimise the objective function for each applied neural network and calculate the corresponding value of the objective function.

Step 5. Implement the neural network model that yields the minimum value of the objective function.

It is shown in the work completed thus far that this procedure is applied to select an optimum neural network model to use for each of the studied industrial applications. Further, it is important to note that the developed and accompanying *Neural Network Analysis and Optimisation Strategy* program facilitates this procedure and therefore simplifies the neural network selection process. It is noted that development of an appropriate optimisation analysis is required as available operations research techniques are inappropriate for neural network optimisation and selection. The optimisation methodology developed is an original strategy that addresses the economic, technical and ergonomic aspects of neural network optimisation and selection. Moreover, while it is highlighted that the developed program is a unique and novel optimisation methodology, it is important to note that it is a tailor made tool that has been carefully developed to facilitate its ease of application to industry. In addition, the accuracy of the developed optimisation methodology is ensured

through rigorous and detailed mathematical studies and meticulous inspection and examination of the analysis results.

While a neural network is selected for implementation for each of the studied industrial applications, it is now interesting to implement the selected neural networks to study process behaviour as a result of the phenomenological modelling. While the expected economic benefits from the application of neural network modelling have been identified in a previous chapter, it is now necessary to quantify these predicted benefits. The results of the neural network implementation for each of the industrial applications are documented in the following chapter.

CHAPTER EIGHT

Neural Network Implementation and Associated Process Improvements

8.1 NEURAL NETWORK IMPLEMENTATION METHODOLOGY

The work documented in this chapter details the procedure and results associated with neural network implementation for each of the studied industrial applications. However, it is important to note that this implementation work is, in the first instance, a validation phase of the modelling methodology. Specifically, while the neural network modelling process thus far has studied the prediction error associated with each applied model, no work has been completed to investigate process behaviour as a result of neural network implementation. Hence, in addition to analysing the capability of each model to accurately predict some performance parameter, it is necessary to study process behaviour as a result of implementing the intelligent decision making logic. For maximum economic benefit and in order to increase the potential to maximise profitability, the neural network implemented in each instance for each of the studied industrial applications is that identified in the optimisation analysis to have the lowest associated prediction cost. Further, only those process parameters nominated in the optimal solution established for each model are used as neural network input variables in each instance.

While the neural network input data has been supplied in the form of a train and test data set during supervised training, this implementation phase requires each model to receive frequent and periodic data to make appropriate predictions of the nominated performance parameter in each instance. The smelter knowledge base contains historical and immediate numerical data for each of the process parameters used as neural network input variables for each of the applications considered. Hence, it is necessary to establish a technique whereby each implemented model is supplied with appropriate input data patterns from the smelter knowledge base, as required. In order

to facilitate data acquisition from the smelter knowledge base it is necessary to develop appropriate programming language to establish communication between the smelter knowledge base and the modelling paradigm. It is useful to note that established techniques exist at CABBL for data acquisition from the knowledge base and the preferred programming language used at the smelter for process control is SAS [231-232]. Hence, the corresponding algorithm and architecture for each specific neural network to be implemented is simulated using SAS programming code. It is useful to note that the technical content and layout of the SAS program code is similar to Pascal and therefore not included as an attachment to the thesis.

While process parameter data is stored on the smelter knowledge base in a broad range of magnitudes corresponding to the particular measurement units for each parameter, it is important to note that each input data vector supplied to a neural network is required to be normalised. It has been shown that the weighted connections developed during neural network training are established for corresponding normalised input-output vectors. Consequently, the magnitude and value of the input data must correspond to the magnitude and value of the training data patterns for which the neural network weights were established. Therefore, the minimum and maximum values used in the normalisation procedure for the train and test data patterns for each application must be used to scale the input data patterns during implementation. Pre-processing of the data patterns extracted from the smelter knowledge base is completed using suitable SAS code. While data acquisition and neural network processing are completed using appropriate SAS code, it is useful to note that the output of the neural network is also interpreted using customised SAS code. It has been noted that the estimate produced by the output layer nodes of each applied neural network is a numerical value in the bounded range 0.0 to 1.0, due to scaling of the output vector in the train and test data sets during supervised training and use of a bounded activation function in the neural network output layer in each instance. Consequently, the weighted connections and activation function in the implemented neural network produce a numerical output in the bounded range 0.0 to 1.0 for a corresponding normalised input vector. Hence, it is necessary to interpret the neural network output values to appropriate units for process control and monitoring. This interpretation for the electrolyte additive and temperature prediction applications is achieved by rearranging the normalising formula given in Equation

2.2.7. Rearranging to solve for x_i yields a formula that can be used to convert a normalised numerical output value from a neural network to its original units. The equation is of the form:

$$x_i = norm(x_i).(\max(x_j) - \min(x_j)) + \min(x_j) \qquad (8.1.1)$$

where, x_i = original i^{th} value in a set of j values,

$norm(x_i)$ = normalised i^{th} value in a set of j values,

$\max(x_j)$ = original maximum value in a set of j values, and

$\min(x_j)$ = original minimum value in a set of j values

In regard to the electrolyte additive prediction application, it has been shown that additions of AlF_3 and Na_2CO_3 to the reduction cell occur in 10.5 and 15.0kg quantities, respectively. Hence, rounding of the output value from the neural network for electrolyte additive predictions is required. This is achieved by rounding the neural network output value, converted to original units using Equation 8.1.1, to the nearest 10.5 or 15.0kg multiple as appropriate for AlF_3 and Na_2CO_3, respectively. For the cell failure prediction application it has been shown that while the neural network output values are in the bounded range 0.0 to 1.0, rounding of the output value to the nearest whole number is required, using Equation 4.3.1. For the electrolyte temperature prediction application requires only conversion of the neural network output to original units is necessary, no rounding is required. Subsequently, the neural network output values from the appropriate interpreter are presented to a user interface such that appropriate process maintenance can be completed. For example, for the electrolyte additive prediction application the estimated quantities of AlF_3 and Na_2CO_3 to add to each particular reduction cell are presented on a user interface such that the appropriate quantity can be added to each cell. Further, for the cell failure prediction application a value of either 0 or 1 is produced for each reduction cell so that a decision can be made whether to remove a particular cell from production. Finally, for the electrolyte temperature prediction application the predicted electrolyte temperature within each reduction cell is displayed on a user interface such that appropriate additions of electrolyte additives can be scheduled to maintain the electrolyte temperature within the predetermined control limits. In addition, the predicted values of the output variables in each instance are recorded on the smelter

knowledge base for future reference. As a result of frequent process control action and due to the dynamic and unstable behaviour of the Hall-Heroult process, the values of parameters associated with the process are continually changing. Specifically, the value of a particular process parameter at time t is not necessarily the same at time t+1, due to evolving process behaviour and process control action. Consequently, periodic measurements of selected process parameters are required, while the data from each measurement is recorded on the smelter knowledge base. Subsequently, this data is used to form the input vectors for neural network predictions as appropriate. The major stages of the neural network implementation methodology are highlighted in Figure 8.1.1.

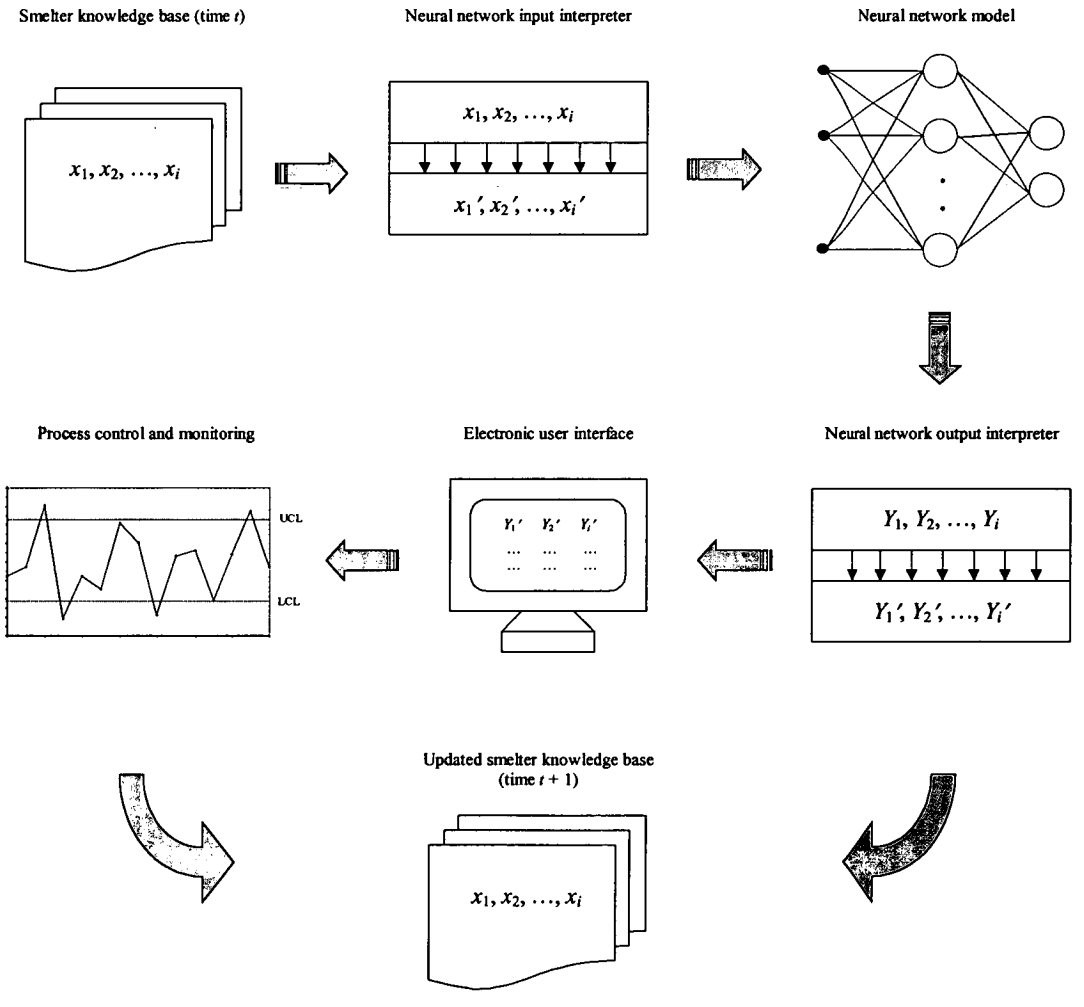


Fig. 8.1.1. Flow Chart Highlighting Major Stages of Neural Network Implementation Procedure

The systematic procedure highlighted is used for neural network implementation for each of the studied industrial applications. It is useful to note that it is typically an

automated procedure in each instance whereby process parameter data is automatically retrieved from the smelter knowledge base and normalised and subsequently, neural network processing is completed automatically. Further, interpretation of the neural network predictions and recording of the output variable values on the smelter knowledge base is completed automatically as a result of the developed SAS code and neural network implementation procedure. In the following documentation the application of this neural network implementation procedure for each of the studied industrial applications is discussed.

8.2 NEURAL NETWORK IMPLEMENTATION FOR ELECTROLYTE ADDITIVE PREDICTION APPLICATION

It has been noted that an important consideration in aluminium smelting is control of reduction cell behaviour. Moreover, minimising electrolyte temperature variation is an important control strategy from an operational point of view. As a means of controlling electrolyte temperature, particular electrolyte additives, namely AlF_3 and Na_2CO_3, are periodically added to each reduction cell. Hence, it is critical that the correct amounts of these additives are scheduled to the cell such that electrolyte temperature is maintained at a desired target value, with minimum deviation from the predetermined target. It is shown that neural networks are applied to predict electrolyte additive additions based on the values of mathematically related measured parameters within the reduction cell. Moreover, it has been shown that the GRNN model incorporating target bath temperature, bath resistivity and temperature reference as input parameters is the most economically beneficial model to use for electrolyte additive prediction. The following work details the performance assessment of the GRNN for predicting the quantity of electrolyte additives to add to the reduction cell. It has been noted that the objective of the neural network for electrolyte additive predictions is to reduce electrolyte temperature variation within the reduction cell while maintaining a mean target temperature of approximately 965.0°C. Moreover, it has been noted that the significant implications of reduced electrolyte temperature variation are improved current efficiency and reduced thermal cycling, resulting in substantial economic benefit in either instance. Further, it is also important to note that the secondary benefit of minimised AlF_3 and Na_2CO_3 wastage is expected as a result of an improved electrolyte additive prediction methodology.

It is important to note that while the objective of neural network modelling in the first instance for this particular application was to accurately predict specified target electrolyte additive quantities, the objective in this instance is to reduce electrolyte temperature variation in the reduction cell. Specifically, while neural network assessment was completed in the training and testing phase by studying the ability of each model to estimate the specified target electrolyte additive quantity, model assessment is completed in this instance by investigating electrolyte temperature behaviour as a result of neural network implementation. While it has been shown that the GRNN is able to accurately predict the correct quantity of AlF_3 and Na_2CO_3 to schedule to the reduction cell, the GRNN model is now applied to investigate its capacity to reduce electrolyte temperature variation. However, it is noted that this assessment could not be completed during the training and testing phase for this application, as it was necessary to firstly select an appropriate model for implementation. Hence, while in the first instance prediction error was assessed by studying the ability of the neural network to achieve target values, the ultimate test of the neural network is to reduce electrolyte temperature variation, which is evaluated as part of this study.

8.2.1 GRNN Implementation Procedure for Electrolyte Additive Prediction Application

In order to investigate the effect of using neural network predictions for scheduling electrolyte additives to the reduction cell it is necessary to nominate some particular reduction cells to be subjected to the modelling strategy. Further, it is necessary to identify and use particular reduction cells that are not subject to alternative experimental examination. Specifically, the reduction cells required for this study must be typical cells that are a good representation of the entire reduction cell population. Consequently, a suitable batch of 22 reduction cells were identified and assigned as test candidates for this neural network implementation investigation. Further, it is important to note that a control group of 21 reduction cells were also monitored during the period of the investigation. The purpose of the control group cells is to provide a means of monitoring process behaviour during the investigation period in similar cells as those in the neural network investigation group. Specifically, the control group cells are used to provide a valid comparison of process behaviour with the neural network controlled cells. Therefore, it is important that the

only difference between the control group and neural network controlled cells is the technique used to estimate the quantity of electrolyte additives to add the reduction cells in either batch. Consequently, the control group cells were carefully selected to satisfy this requirement. The reduction cells corresponding to electrolyte additive additions based on neural network predictions are classified as *Cell Group A*, while the reduction cells corresponding to electrolyte additive additions based on the existing regression analysis, or control group cells, are classified as *Cell Group B*.

The duration of the investigation was 30 days. Specifically, the developed GRNN model was implemented and used to make electrolyte additive predictions for a period of 30 days. During this period, the electrolyte temperature was periodically measured in the reduction cells in *Cell Group A* and *Cell Group B*. It is important to note that the electrolyte additive scheduling methodology was not altered for the reduction cells in *Cell Group B*, the previously described regression analysis technique was maintained. Further, in order to make an assessment of the affect of the neural network modelling methodology on process control it was necessary to observe electrolyte temperature behaviour prior to neural network implementation. Hence, the electrolyte temperature in the reduction cells in *Cell Group A* and *Cell Group B* for the 30 day period prior to neural network implementation was observed. Thus, electrolyte temperature behaviour in the reduction cells prior to and following neural network implementation was studied to assess the effect the neural network implementation had on electrolyte temperature behaviour. Further, the effect of neural network modelling was inspected by observing electrolyte temperature behaviour in the control group cells to quantify whether the exhibited process behaviour was due to the neural network implementation or due to normal process behaviour.

It has been noted that electrolyte temperature measurements are completed on a bi-daily frequency, or every 48.0 hours, hence, a total of 15 electrolyte temperature measurements are recorded for each reduction cell in either cell group over the 30 day investigation period. From the electrolyte temperature measurements, the mean, or average, electrolyte temperature and standard deviation of electrolyte temperature in each reduction cell in either batch was determined for the 30 day investigation

period. Specifically, the mean electrolyte temperature, μ, is calculated using the following formula:

$$\mu = \frac{\sum_{1}^{i} X_i}{n} \tag{8.2.1}$$

where, X_i = electrolyte temperature i,

$i = 1, ..., n$, and

n = number of electrolyte temperatures in the population

Electrolyte temperature variation is assessed through the standard deviation, σ, of temperature measurements. Standard deviation is a measure of how widely values from a population are dispersed from the population mean and is calculated using the following formula [233]:

$$\sigma = \sqrt{\frac{\sum_{1}^{i} (X_i - \mu)^2}{n}} \tag{8.2.2}$$

where, $\sum_{1}^{i} (X_i - \mu)^2$ = sum of squared deviations between each population value, X_i, and the population mean, μ,

$i = 1, ..., n$, and

n = number of electrolyte temperatures in the population

Hence, standard deviation is a measure of electrolyte temperature dispersion from the mean electrolyte temperature. Further, standard deviation is proportional to electrolyte temperature variation. Specifically, the higher the magnitude of electrolyte temperature deviation from the mean, then higher is the standard deviation. Hence, the desired outcome of neural network implementation in this instance is to observe lower standard deviation of electrolyte temperature in the reduction cell subsequent to neural network implementation, while achieving a mean electrolyte temperature of approximately 965.0°C.

In order to accurately and quantitatively compare the mean and standard deviation of electrolyte temperature in each reduction cell prior and subsequent to neural network implementation a statistical analysis technique is applied. While the *t*-statistic test is used to compare the population means, an alternative statistical test is required to compare the population variances, or standard deviation. The statistical *f*-test [234-236] is based on a probability distribution and is appropriate in this instance for comparison of the standard deviation of electrolyte temperature in either group of cells. The null hypothesis, H_o, for the *f*-test is that the standard deviation of the populations are equal, while the alternative hypothesis, H_a, is that the standard deviation of the populations are unequal. Similar to the *t*-test, the null hypothesis is accepted if the test statistic is lower than some predetermined critical value, otherwise, the alternative hypothesis is accepted. An *f*-critical value is determined by three parameters; the significance level, σ, the numerator degrees of freedom, ν_1, and the denominator degrees of freedom, ν_2. While a significance level of 0.05 is typically used for the statistical *f*-test, the value ν_1 is equivalent to n_1-1, where n_1 is the number of observations in the first population, while ν_2 is equivalent to n_2-1, where n_2 is the number of observations in the second population.

8.2.2 GRNN Implementation Results for Electrolyte Additive Prediction Application

The mean and standard deviation of electrolyte temperature prior and subsequent to neural network implementation is presented in Tables 8.2.1 and 8.2.2 for *Cell Group A* and *Cell Group B*, respectively. It is necessary to note that the 30 day period prior to neural network implementation is distinguished in the results as (1) while the 30 day period following neural network implementation is distinguished in the results as (2). Hence, the mean electrolyte temperature is noted before and after neural network implementation, μ (1) and μ (2), respectively. Similarly, the standard deviation of electrolyte temperature is noted before and after neural network implementation, σ (1) and σ (2), respectively. In addition, it is shown that a *t*-statistic value is calculated for the mean electrolyte temperature, while an *f*-statistic value is determined for the standard deviation of electrolyte temperature. In each instance, it is necessary to determine a critical statistic value. For the *t*-statistic test it has been shown that the critical value is determined by the significance level and the degrees of freedom

associated with the population data. Hence, for a significance level of 0.01 and degrees of freedom of 14, as there are 15 observations for each cell each population in period's (1) and (2), a *t*-critical value of 2.970 is appropriate. In regard to the *f*-test, the parameters ν_1 and ν_2 are equal to 14 in either instance, while a significance level of 0.05 yields an *f*-critical value of 2.491.

TABLE 8.2.1. Mean Electrolyte Temperature, μ, and Standard Deviation, σ, for Electrolyte Additive Prediction Application (*Cell Group A*)

Cell Number	μ (1) (°C)	μ (2) (°C)	*t*-statistic	σ (1) (°C)	σ (2) (°C)	*f*-statistic
01	963.3	965.7	0.892	12.63	7.30	2.991
02	968.2	963.1	1.953	7.49	9.88	1.743
03	966.7	967.6	0.281	12.58	7.66	2.705
04	966.6	968.9	0.590	11.23	7.31	2.361
05	965.9	966.9	0.323	9.50	8.24	1.336
06	965.3	967.4	0.348	12.50	11.72	1.148
07	967.5	965.5	0.719	10.18	7.93	1.655
08	965.4	966.8	0.568	11.06	9.55	1.340
09	963.5	962.7	0.644	12.28	7.86	2.444
10	966.7	963.5	0.335	9.47	9.07	1.096
11	967.4	964.8	0.227	10.76	9.56	1.272
12	964.1	963.5	0.797	8.72	8.37	1.094
13	963.4	965.9	1.011	8.65	12.12	1.961
14	964.5	965.1	0.204	11.67	10.93	1.142
15	965.9	971.3	0.643	11.08	8.75	1.600
16	965.0	965.8	0.122	10.08	12.92	1.645
17	965.5	965.8	1.397	11.07	9.78	1.287
18	964.9	969.3	0.151	10.09	11.47	1.293
19	962.5	964.7	0.059	9.81	13.56	1.911
20	968.4	967.8	0.914	12.76	11.18	1.308
21	965.8	965.6	0.463	9.14	9.57	1.108
22	963.8	967.3	0.136	15.41	11.86	1.692
Average	965.5	966.1	-	10.83	9.85	-

It is shown that the mean electrolyte temperature prior and subsequent to neural network implementation is not significantly statistically different in either instance for any of the reduction cells in *Cell Group A*. Specifically, the null hypothesis is accepted in each instance as the *t*-statistic is lower than the established *t*-critical value. Overall, it is shown that the average electrolyte temperature in *Cell Group A* is 965.5°C for the 30 day period prior to neural network implementation, which is not statistically significantly different to the mean electrolyte temperature of 966.1°C for the 30 day period following neural network implementation. Considering the

standard deviation of electrolyte temperature, it is shown that electrolyte temperature variation reduced in 2 of the reduction cells in which electrolyte additives where scheduled using the developed GRNN model. It is shown that the standard deviation of electrolyte temperature in cell 01 reduced from 12.63 prior to neural network implementation to 7.30 subsequent to implementation, confirmed to be statistically significant. Further, the standard deviation of electrolyte temperature in cell 03 exhibited a statistically significant reduction from 12.56 to 7.66 prior and subsequent to neural network implementation, respectively. However, it is shown for the remainder of the reduction cells in *Cell Group A* that the standard deviation of electrolyte temperature before and after neural network implementation is not statistically significantly different. It is shown for each remaining cell that the null hypothesis is accepted in each instance.

TABLE 8.2.2. Mean Electrolyte Temperature, μ, and Standard Deviation, σ, for Electrolyte Additive Prediction Application (*Cell Group B*)

Cell Number	μ_{temp} (1) (°C)	μ_{temp} (2) (°C)	*t*-statistic	σ (1) (°C)	σ (2) (°C)	*f*-statistic
01	964.2	964.7	0.218	8.76	11.85	1.831
02	965.6	965.6	0.057	7.67	11.09	2.090
03	966.3	968.7	1.073	10.48	12.16	1.353
04	963.3	965.1	0.894	8.40	12.25	2.133
05	968.4	971.1	1.216	6.03	12.35	4.196
06	961.8	961.7	0.112	8.06	7.86	1.052
07	963.5	966.5	0.950	6.96	9.09	1.719
08	965.5	966.7	0.438	5.58	12.50	5.021
09	964.1	967.8	0.842	6.52	9.16	1.978
10	962.3	963.1	0.149	6.95	10.82	2.420
11	967.9	969.8	0.834	7.30	10.62	2.128
12	964.3	967.6	1.341	5.59	10.78	3.724
13	965.2	968.6	1.094	8.23	9.69	1.385
14	964.1	966.2	1.106	8.99	10.56	1.382
15	962.9	963.9	0.725	8.18	9.00	1.218
16	964.7	963.6	0.638	5.00	9.04	3.281
17	964.8	966.1	1.214	9.28	9.25	1.019
18	962.5	964.6	0.784	8.26	9.66	1.375
19	962.1	962.3	0.092	10.35	8.09	1.648
20	965.3	967.7	0.561	10.65	8.55	1.553
21	964.8	963.4	0.752	7.16	6.84	1.107
Average	964.5	965.9	-	7.83	10.06	-

It is shown for the control group cells that the mean electrolyte temperature is not statistically significantly different in period's (1) and (2). In each instance it is shown

that the null hypothesis is accepted as the *t*-statistic is lower than the *t*-critical value for each cell. Hence, the average electrolyte temperature of 964.5°C for the reduction cells in *Cell Group B* during period (1) is not statistically significantly different to the mean electrolyte temperature of 965.9°C calculated for period (2). This is an important finding as it confirms that standard process control during the investigation period did not yield an increase or decrease in mean electrolyte temperature and consequently, the neural network controlled cells should not exhibit an increase or decrease in mean electrolyte temperature. However, it is shown that 4 of the reduction cells in *Cell Group B* exhibited an increase in electrolyte temperature variation during the investigation period. In particular, cell's 05, 08, 12 and 16 have each shown an *f*-statistic that is higher than the noted *f*-critical value, accepting the alternative hypothesis that the population variances are unequal. Further, it is shown in each instance for these 4 reduction cells that the standard deviation of electrolyte temperature increases from period (1) to (2). This finding indicates that standard process control during the investigation period influenced electrolyte temperature to exhibit higher variation in some reduction cells. On the other hand, the remainder of the reduction cells in *Cell Group B* have shown standard deviation values that are confirmed to be not statistically significantly different in each instance. Hence, it is interesting to note that while the reduction cells in the neural network controlled group exhibited reduced or equivalent electrolyte temperature variation from period (1) to (2), the reduction cells in the control group have exhibited increased or equivalent standard deviation of electrolyte temperature from period (1) to (2). However, while it is not possible to quantify whether electrolyte temperature variation would have increased in some of the reduction cells in *Cell Group A* if the GRNN had not been applied, it is appropriate to make some assumptions. As the control group cells are identical to the neural network controlled cells and due to the fact that the process control procedures administered to each batch are identical then it is probable that some of the reduction cells in *Cell Group A* would have exhibited increased electrolyte temperature variation if the GRNN had not been applied, as a result of standard process control. Hence, it can be assumed that the neural network control strategy succeeded in reducing electrolyte temperature variation.

In addition to observing electrolyte temperature behaviour corresponding to implementation of the GRNN, it is also useful to study electrolyte additive usage as a

result of neural network modelling. The quantity of AlF_3 and Na_2CO_3 used in the 30 day period prior to and 30 day period subsequent to neural network implementation, again denoted as (1) and (2), respectively, is shown in Table 8.2.3. In regard to *Cell Group A*, it is shown that AlF_3 usage prior to neural network implementation is 10.4kg per cell per 24.0 hour period, or per day, while Na_2CO_3 usage is at a rate of 1.1kg per cell per day. Likewise, AlF_3 and Na_2CO_3 usage is the same for reduction cells in *Cell Group B* during period (1). However, while AlF_3 and Na_2CO_3 usage is shown to remain the same during period (2) in *Cell Group B*, it is shown that AlF_3 and Na_2CO_3 usage decreases significantly in *Cell Group A* subsequent to neural network implementation. In particular, it is shown that AlF_3 usage reduces to 9.6kg per cell per day, while Na_2CO_3 usage is at a reduced rate of 0.3kg per cell per day following neural network implementation. It has been previously noted that the majority of Na_2CO_3 that is added to the reduction cell is required to compensate for surplus AlF_3 additions. Hence, the majority of Na_2CO_3 is wasted due to incorrect AlF_3 additions using the existing prediction technique. However, with more accurate predictions of AlF_3 in the first instance using the developed GRNN model, while reducing AlF_3 usage, Na_2CO_3 usage is also substantially reduced.

TABLE 8.2.3. AlF_3 and Na_2CO_3 Usage per Reduction Cell per 24 Hour Period for Electrolyte Additive Prediction Application

	Electrolyte Additive Usage per Cell per Day (kg)	
Additive (Period)	*Cell Group A*	*Cell Group B*
AlF_3 (1)	10.4	10.4
Na_2CO_3 (1)	1.1	1.1
AlF_3 (2)	9.6	10.4
Na_2CO_3 (2)	0.3	1.1

8.2.3 Process Improvements Associated with GRNN Implementation for Electrolyte Additive Prediction Application

There are specific economic, ergonomic and technical improvements achievable at CABBL as a direct result of the implementation of the GRNN model for electrolyte additive prediction. The following discussion highlights the particularities of each of these improvements.

Economic - Two significant findings with major implications in regard to process behaviour have resulted from neural network modelling for this particular

application. In the first instance, it is shown that electrolyte temperature variation in the reduction cell is lower than that achieved using the existing electrolyte additive prediction technique, while maintaining an average electrolyte temperature of approximately 965.0°C. It has been noted that the economic benefit of reduced temperature variation is attributed to higher efficiency and reduced thermal cycling of the reduction cell, leading to extended cell production life. The economic value of improved current efficiency and extended cell production life resulting from neural network implementation is estimated to be equivalent to approximately $1,000.00 per reduction cell per annum. As there is typically an average of 540 reduction cells in production at any instant at CABBL then a potential annual saving of $540,000.00 is achievable as a result of neural network implementation for electrolyte additive prediction. In addition, electrolyte additive usage is shown to be lower using the neural network modelling technique, which has also been shown to yield an economic benefit. Specifically, it is noted that approximately 0.8kg of AlF_3 and Na_2CO_3 per cell per 24.0 hour period is saved as a result of the neural network electrolyte additive prediction strategy. For a typical production facility of 540 cells, the electrolyte additive saving equates to approximately 157.7 tonnes of AlF_3 and Na_2CO_3 per annum. Typical costs for AlF_3 and Na_2CO_3 are at a rate of $1,500.00 and $700.00, respectively, yielding an annual saving of $236,550 for AlF_3 and $110,390 for Na_2CO_3. Hence, the economic benefit associated with neural network implementation for electrolyte additive prediction is summarised as follows:

Current efficiency:	$540,000.00
AlF_3 usage:	$236,550.00
Na_2CO_3 usage:	$110,390.00
	$886,940.00 TOTAL

Hence, neural network implementation for electrolyte additive prediction yields a potential saving of approximately $886,940.00 per annum. This saving represents a substantial reduction in production costs at the smelter and is a direct result of neural network implementation for electrolyte additive prediction.

Ergonomic - An ergonomic benefit resulting from the use of neural networks for electrolyte additive prediction is reduced operator exposure to the reduction cell

environment. In particular, it has been shown that AlF_3 and Na_2CO_3 usage is significantly reduced as a result of neural network implementation. Further, it has been noted that AlF_3 and Na_2CO_3 additions to the reduction cell are completed manually. It is shown in Figure 4.2.5(a) that AlF_3 is added manually via a storage hopper while Figure 4.2.5(b) shows bags of Na_2CO_3 being added manually to the reduction cell. Hence, the manual addition of these important electrolyte additives requires operators to work in the reduction cell environment for a time period that is proportional to the quantity of electrolyte additives to add. Consequently, a reduction in electrolyte additive quantity results in reduced operator exposure to the harsh potline environment. This is particularly relevant to the addition of Na_2CO_3, which is physically demanding of process operators. The substantial reduction in AlF_3 and Na_2CO_3 usage directly results in reduced operator exposure to the reduction cell environment and reduced physical labour requirement, which are both significant process improvements and are a result of neural network implementation.

Technical - The adoption of neural networks for modelling of the Hall-Heroult process is a significant technical improvement at CABBL. It is interesting to note that the application of neural networks for electrolyte additive prediction is the first neural network modelling application to be investigated at CABBL. However, the neural network performance exhibited in this instance gives confidence in applying this modelling strategy to further applications at the smelter. Hence, the technical improvement in this instance is that a new modelling methodology has been identified at CABBL and necessary integration of the modelling paradigm into the smelter knowledge base has been established, facilitating further application of neural networks at the smelter.

8.3 NEURAL NETWORK IMPLEMENTATION FOR CELL FAILURE PREDICTION APPLICATION

The ability to accurately estimate whether a reduction cell should be removed from operation or maintained in production has been highlighted as a critical consideration for lowering aluminium production costs. In addition, it has been shown that empirical modelling using neural networks is appropriate to identify reduction cells that have high potential for failure in a predetermined period. Moreover, it has been highlighted that the WH neural network provides the most economically beneficial

process model that can be applied to determine whether a reduction cell will fail during the prediction period. The process parameters nominated in the optimal solution for the WH model to make cell failure predictions include lining voltage drop, cell age, anode effect frequency and duration, unscheduled anode change, cell power, high frequency noise, cell resistance, bath resistivity, cell voltage, anode effect energy and high temperature count. It is useful to note that the objective of the neural network implementation study in this instance is to investigate the ability of the developed WH model to accurately predict which particular reduction cells are high risk failure candidates without compromising cell production life. Specifically, this investigation is used to confirm the WH network prediction accuracy established using the test data set for the cell failure prediction application. The following discussion details the procedure used in this instance and the results of implementing the WH neural network for cell failure prediction.

8.3.1 WH Implementation Procedure for Cell Failure Prediction Application

The duration of the validation investigation was 35 days in this instance. During this time the WH neural network was programmed to make cell failure predictions on a weekly basis. Hence, during the validation period, 5 predictions were made for each reduction cell in a specific cell population. In particular, the nominated reduction cell group used for the study in this instance consisted of 200 cells. Consequently, a total of 1,000 cell failure predictions were obtained during the validation study. In order to quantify the accuracy of the predictions the reduction cells in the study group were not removed from production if predicted to fail by the WH neural network. Consequently, observing whether the identified cell did indeed fail assessed the accuracy of a failure prediction. Further, the number of days the cell remained in production after an incorrect failure prediction provided valuable information on the accuracy of the neural network model. On the other hand, reduction cells in the study population that did fail during the investigation period that were not identified by the neural network as failure candidates also assessed the accuracy of the developed WH model. Hence, at the conclusion of the investigation period, the neural network predictions were compared with actual cell behaviour. The results of this comparison are documented in the following section.

8.3.2 WH Implementation Results for Cell Failure Prediction Application

During the period of the investigation a total of 7 reduction cells in the study population size of 200 failed and were consequently removed from production. The particular reduction cells that failed during the validation study are identified by a cell number, as shown in Table 8.3.1. Further, the prediction values obtained for each of the 5 predictions during the validation study for each of the 7 failed reduction cells are shown. It is useful to note that the non-interpreted prediction value is documented in the table, rather than a rounded value of either 0 or 1. In addition, Prediction 1, 2, 3, 4 and 5 correspond to the neural network predictions in the first, second, third, fourth and fifth week, respectively, of the investigation.

TABLE 8.3.1. WH Neural Network Cell Failure Predictions During Investigation Period

Cell No.	Prediction 1	Prediction 2	Prediction 3	Prediction 4	Prediction 5
108	0.99 *	0.00	0.00	0.00	0.00
152	0.86	0.94 *	0.00	0.01	0.00
057	0.29	0.27 *	0.00	0.00	0.00
066	0.03	0.22	0.98 *	0.00	0.00
083	0.08	0.12	0.37	0.96 *	0.00
162	0.00	0.00	0.00	0.15	0.97 *
184	0.00	0.00	0.19	0.38	0.99 *
078	0.00	0.58	0.14	0.02	0.02
119	0.73	0.11	0.00	0.00	0.00

* indicates reduction cell failure and removal from production

It can be seen that cell 108 was predicted by the WH model to fail in the first week of the study and subsequently did fail and was removed from production. Similarly, cell 152 was identified in the second week of the investigation as a high risk failure candidate, denoted by a prediction value of 0.94, which indeed did fail in the second week of the investigation and was consequently removed from production. However, it is shown that the prediction in the first week of the study (Prediction 1) for cell 152 produced a value of 0.86, which also indicates high failure risk. Consequently, in a full-scale implementation situation cell 152 would have been removed from production in this instance and not allowed to remain in production for the extra week, resulting in a week of lost production from the reduction cell, which has been shown to have an associated economic disadvantage. On the other hand, cell 057 is shown to have failed in the second week of the validation study, however, the neural

network did not predict the failure. The predicted value of 0.27 noted for cell 152 in the second week of the investigation is not sufficient to remove the cell from production. It has been noted that only a value of 0.50 or greater represents high risk failure candidates. Nevertheless, it is shown that cells 066, 083, 162 and 184 were correctly identified as failure candidates immediately prior to the actual failure occurrence. It is shown that cell 066 was correctly identified to fail in the third week of the study, cell 083 was accurately predicted to fail in the fourth week of the investigation, while cells 162 and 184 were correctly identified to fail in the fifth week of the validation study. On the other hand, cells 078 and 119 are shown to have prediction values greater than 0.50, although it is shown that these cells did not fail during the investigation period. Specifically, it is shown that cell 078 has a predicted value of 0.58 in the second week of the study, highlighting this cell as a high risk failure candidate, however, this cell did not fail during the investigation period. Similarly, cell 119 exhibited a prediction of 0.73 in the first week of the investigation, although the cell did not fail during the investigation period. However, it is interesting to note that the predicted values for these cells decreased with increasing time. In particular, in the third week of the study, cell 078 showed a prediction of 0.14, highlighting the cell as a non-failure candidate, while cell 119 produced a prediction of 0.11 in the second week of the investigation, again highlighting the cell as a non-failure candidate. It is interesting to note that while a study of the input vectors for these cells corresponding to the incorrect predictions was completed, there was no parameter that was obvious as having an extreme value that would influence the WH model to incorrectly predict either cell as a failure candidate. Rather, it is a unique combination of the input variables that yields a high prediction and it is not obvious from manual observation what this combination is.

While predictions associated with failed reduction cells and incorrect failure predictions are noted in the table, it is important to note that the remaining cells in the study population obtained predicted values of approximately zero for each of the 5 predictions during the validation study. Attributed to the fact that the remaining reduction cells in the study population did not fail during the investigation period, it follows that the WH predictions were correct for the remaining 191 cells not documented in the preceding table. This particular finding has major implications in regard to implementation of the developed WH model for cell failure predictions as it

highlights the ability of the WH neural network to correctly identify cells that are not failure candidates. It has been noted that this is an important consideration as premature cell removal yields an economic disadvantage at a rate of approximately $34.00 per cell per 24.0 hour period.

Nevertheless, the WH model has shown high accuracy for cell failure prediction. In particular, in regard to the 7 reduction cells that failed during the validation study, 5 cells were correctly identified as failure candidates immediately prior to the failure occurrence (108, 066, 083, 162, 184). However, 1 reduction cell, while correctly identified as a failure candidate immediately prior to the failure occurrence, was also incorrectly identified as a failure candidate one prediction period prior to the failure occurrence (152), while the remaining cell that failed during the validation study was not identified as a failure candidate (057). On the other hand, 2 cells were predicted as failure candidates although they did not fail during the validation study period (078, 119). It is also noted that the remaining cells in the study population were correctly identified as non-failure candidates. Hence, it can be seen that from the 1,000 neural network predictions completed during the validation study, only 4 incorrect predictions were made, representing a prediction error rate of 0.4%. Consequently, assuming the neural network accuracy exhibited during the validation study is maintained it can be stated that the WH neural network for cell failure prediction has an accuracy of 99.6%, which is exceptionally high considering the dynamic and unstable behaviour of the Hall-Heroult process.

8.3.3 Process Improvements Associated with WH Implementation for Cell Failure Prediction Application

Implementation of the WH neural network model for cell failure prediction has resulted in specific economic, ergonomic and technical improvements at CABBL. The following discussion highlights each of these improvements.

Economic - The economic benefit associated with cell failure prediction using neural networks is attributed to reduced tap-out occurrences. It has been noted that a tap-out occurrence at CABBL costs approximately $2,500. Moreover, the frequency of tap-outs at the smelter is at a rate of approximately 42 per annum. Hence, the cost associated with tap-out occurrences at CABBL is approximately $105,000 per

annum. This is the economic value of the potential saving achievable using neural networks for cell failure prediction. It is noted that the majority of this potential saving is achievable using the WH model as a prediction accuracy of 99.6% is obtained. Hence, neural network implementation in this instance has the potential to substantially reduce production costs at the smelter.

Ergonomic - The ergonomic benefit associated with the implementation of neural networks in this instance is attributed to using the empirical modelling technique as a preventative maintenance strategy. Specifically, cell failure prediction and removal prior to tap-out eliminates the requirement for operators to enter the reduction cell surroundings to complete arduous and dangerous manual tasks. This is a significant implication as minimising operator exposure to the reduction cell environment is a desirable objective in the aluminium smelting industry.

Technical - The technical benefit arising from cell failure prediction using neural networks is through improved understanding of mechanisms that lead to premature cell failure. For instance, it has been noted that anode effect frequency, unscheduled anode changes and high temperature excursion frequency have some correlation with cell failure in most instances. These particular process parameters are associated with cell control and are affected by operator handling of the reduction cell and cell control strategies. Specifically, it is noted that the way in which the reduction cells are maintained has a significant correlation with cell life. Hence, the neural network modelling in this instance has highlighted the importance of high quality cell control strategies for extended cell life.

8.4 NEURAL NETWORK IMPLEMENTATION FOR ELECTROLYTE TEMPERATURE PREDICTION APPLICATION

It has been noted that the objective of neural network modelling in this instance is to provide an intelligent process modelling technique whereby accurate predictions of electrolyte temperature are obtained. The rationale for the requirement for an electrolyte temperature prediction methodology is a reduction in aluminium production costs. It has been noted that a cost reduction is achievable through reduced manual labour requirement, equipment consumption and associated injuries. It has been shown that the most economically beneficial neural network to use for the

electrolyte temperature prediction application is the BP2 model. Further, it has been shown that the process parameters required as input variables for the BP2 model are electrolyte temperature (t-1), bath resistivity, electromotive force, cell age, bath height, high and low frequency noise, cell power and AlF_3 and Na_2CO_3 addition. The validation procedure outlined in the following section highlights that neural network implementation in this instance is required to study the capability of the developed BP2 model to make accurate predictions of electrolyte temperature. While the test data set used for this application has shown relatively accurate neural network predictions of electrolyte temperature using the BP2 model, it is necessary to complete this study to confirm the accuracy and generalisation ability of the developed model.

8.4.1 BP2 Implementation Procedure for Electrolyte Temperature Prediction Application

The duration of the investigation in this instance was 30 days. During this period the manual electrolyte temperature measurement procedure detailed in Chapter Four was maintained, while electrolyte temperature predictions were completed using the BP2 neural network. Consequently, the BP2 model predicted values were compared with the manually measured values to identify any discrepancy. The nominated reduction cells used for neural network implementation in this instance constitute a particular section of the potline in which no other experimental or development work was being completed. In particular, 105 reduction cells were used for the investigation. It is necessary to note that the cell population used for this investigation were representative of the total reduction cell population; they were standard reduction cells. During the investigation period a total of 15 manual electrolyte temperature measurements were recorded for each reduction cell in the study population, as a consequence of the 48.0 hour manual electrolyte temperature measurement frequency. Consequently, the BP2 neural network was programmed to make electrolyte temperature predictions at the particular time that corresponded with the manually measured electrolyte temperatures. Hence, for each reduction cell in the study population, the 15 neural network predicted electrolyte temperature values over the 30 day period were compared with the 15 manual electrolyte temperature measurements obtained. Hence, the comparison completed in this instance studies the deviation of the predicted electrolyte temperature value from the actual measured

electrolyte temperature. Consequently, the analysis completed in this instance studies the ability of the BP2 neural network to reproduce the prediction accuracy exhibited for the test data set when previously unseen input vectors are presented to the model. It is important to note that as a result of using 105 reduction cells for the validation study and further, obtaining 15 electrolyte temperature measurements and corresponding predictions per cell, a total of 1,575 electrolyte temperature values were available for comparison in this instance, giving confidence in the analysis results.

8.4.2 BP2 Implementation Results for Electrolyte Temperature Prediction Application

In the first instance it is interesting to observe a dispersion plot of actual and predicted electrolyte temperature values, as shown in Figure 8.4.1.

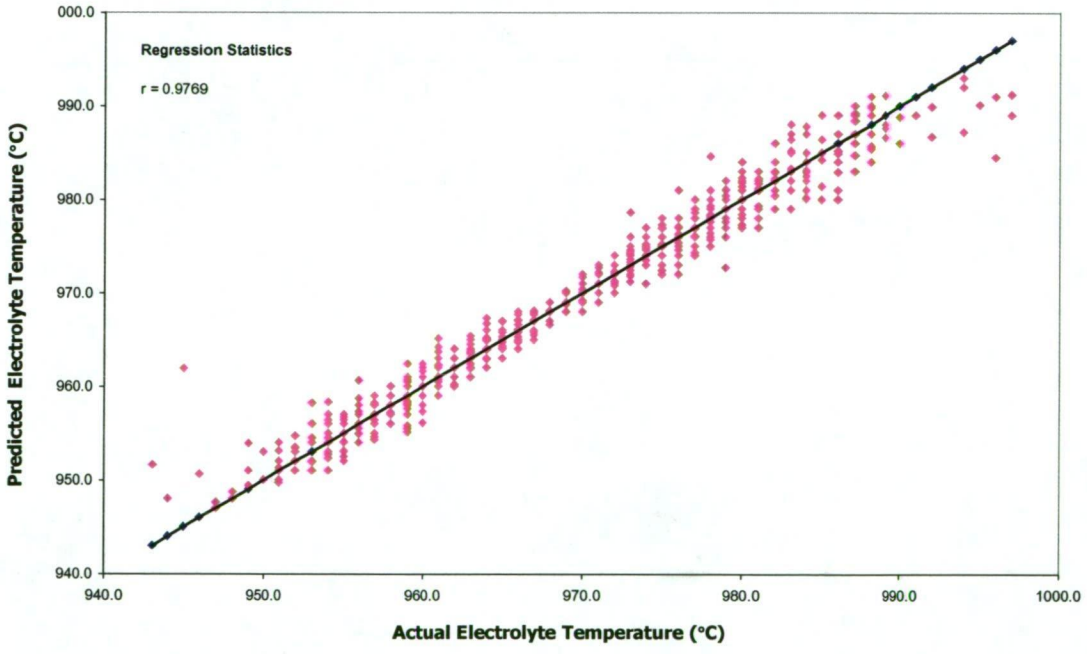


Fig. 8.4.1. Dispersion Plot for GRNN Model for Electrolyte Temperature Prediction Application

While the correlation coefficient, *r*, of 0.9769 noted on the dispersion plot indicates a high correlation between the actual and predicted electrolyte temperature values, it is shown that the predicted values have a higher dispersion at the extremes of the electrolyte temperature range. Specifically, it is shown that deviation of the predicted electrolyte temperature from the actual value is higher for temperatures lower than

approximately 945.0°C and higher than approximately 985.0°C. However, it has been noted that this particular behaviour is expected as extremely low and high electrolyte temperatures were not well represented in the training data patterns, due to the lack of available data from the smelter knowledge base. Nevertheless, it is shown that for the majority of manual electrolyte temperature measurements completed, the corresponding predicted values were of high accuracy, confirmed by the high value of the correlation coefficient.

While a dispersion plot is useful to study the accuracy of the BP2 neural network it is also interesting to produce an error histogram for the model to determine whether the model is biased towards higher-than-actual or lower-than-actual predictions. However, it is shown in Figure 8.4.2 that an approximately normal distribution of prediction error is associated with the BP2 model.

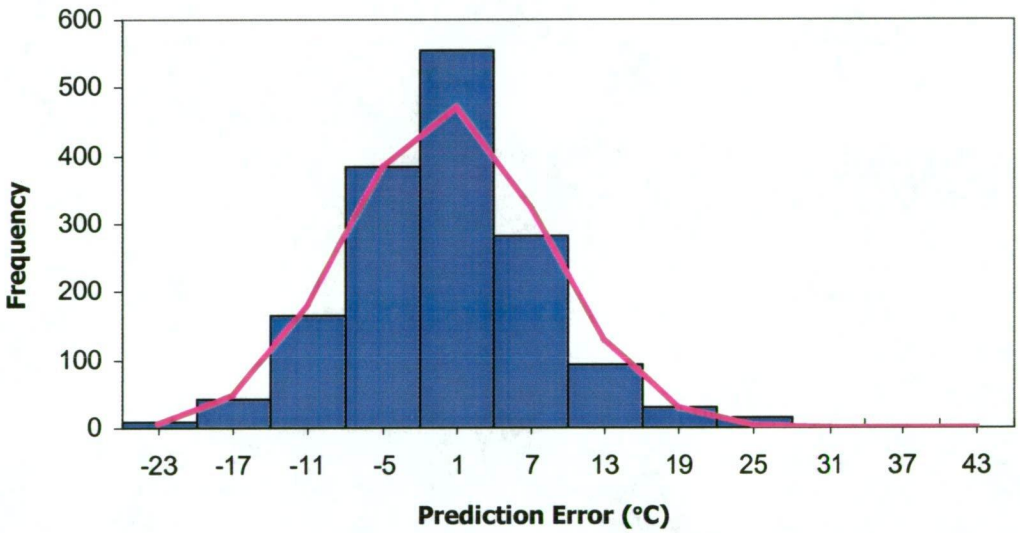


Fig. 8.4.2. Histogram Showing Prediction Error Distribution for BP2 Model for Electrolyte Temperature Prediction Application

It can be seen that while a substantial number of predictions have approximately zero error, the normal distribution approximation curve highlights an approximately equal distribution of predictions that are higher or lower than the actual measured electrolyte temperature values, confirmed by the descriptive statistics of the histogram, noted in Table 8.4.1. It is shown that the mean value and skewness of the histogram are 0.04°C and 0.02°C, respectively. Further, it is noted that approximately

50.0% of the prediction error is lower than 0.0°C, again confirming the normal distribution of prediction error. This is an important finding as it highlights the BP2 model as not biased towards higher-than-actual or lower-than-actual predictions.

TABLE 8.4.1. Prediction Error Histogram Statistics for BP2 Model for Electrolyte Temperature Prediction Application

Descriptive Statistics	Statistical Value
data size	1,575
minimum value (°C)	-11.54
maximum value (°C)	16.90
range (°C)	28.44
mean value, or 50th percentile (°C)	0.04
standard deviation (°C)	7.83
skewness (°C)	0.02
proportion of error below 0.0°C	0.50

The average prediction error associated with each of the nominated electrolyte temperature ranges is shown in Table 8.4.2. It can be seen that the average prediction error associated with each range is very similar in each instance to the average prediction error associated with the same electrolyte temperature range for the test data set, as documented in Chapter Five. In particular, it is shown that the minimum prediction error is associated with the temperature range 965.1°C to 975.0°C, shown to be approximately ±1.97°C. In addition, it is shown that approximately 90.0% of electrolyte temperature measurements are in the range 955.1°C to 985.0°C, which has an associated weighted average prediction error of ±2.44°C. It is useful to note that the weighted average prediction error is calculated by multiplying the average prediction error associated with each temperature range by the proportion of temperatures in that range.

TABLE 8.4.2. Prediction Error Associated with Specified Data Ranges for BP2 Model for Electrolyte Temperature Prediction Application

Bath Temperature Range (°C)	Average Prediction Error in this Range (°C)	Percentage of Temp. in this Range (%)
935.1 to 945.0	11.03	0.4
945.1 to 955.0	4.52	8.6
955.1 to 965.0	2.68	45.4
965.1 to 975.0	1.97	32.5
975.1 to 985.0	4.98	11.8
985.1 to 995.0	12.14	1.3

Hence, the BP2 neural network in this instance has shown comparable results to that obtained for the test data set, confirming the generalisation ability of the developed model. It is shown for previously unseen data patterns presented to the neural network that the BP2 model is able to interpolate to make accurate predictions of electrolyte temperature.

8.4.3 Process Improvements Associated with BP2 Implementation for Electrolyte Temperature Prediction Application

Similar to the previous applications, implementation of the BP2 neural network model for electrolyte temperature prediction has resulted in specific economic, ergonomic and technical improvements at CABBL. It is useful to discuss each of these improvements and highlight their significance in regard to aluminium smelting at CABBL.

Economic - The major economic benefit resulting from neural network implementation for electrolyte temperature prediction is attributed to reduced labour requirement. It has been noted that the labour requirement associated with bath temperature measurement is at a rate of approximately 1,779.00 hours per annum, at a cost of approximately $35.00 per hour. Hence, a potential labour saving of approximately $62,265.00 per annum is achievable if manual bath temperature measurement is eliminated at the smelter. A further economic benefit to result from the application of neural networks for electrolyte temperature prediction is reduced equipment consumption. The corrosive environment within the reduction cell yields deterioration and ultimate failure of the thermocouples used for electrolyte temperature measurement. The cost associated with thermocouple consumption is at a rate of approximately $30,000.00 per annum. In addition, injuries associated with electrolyte temperature measurement yield annual costs of approximately $1,000.00. Hence, the economic benefit associated with neural network implementation for electrolyte temperature prediction is summarised as follows:

Labour:	$62,265.00	
Equipment:	$30,000.00	
Injuries:	$ 1,000.00	
	$93,265.00	TOTAL

Hence, neural network implementation for electrolyte temperature prediction yields a potential saving of approximately $93,265.00 per annum. This potential saving is a direct consequence of neural network implementation for electrolyte temperature prediction.

Ergonomic - The ergonomic benefit resulting from inferential temperature measurement using neural networks is reduced operator exposure to the harsh reduction cell environment. As a result of neural network implementation there is no longer a requirement for operators to be exposed to the high temperature, dusty and dangerous reduction cell surroundings for electrolyte temperature measurement. It has been noted that minimising operator exposure to the harsh and unstable reduction cell environment is a major objective of aluminium smelting process improvements. Hence, neural network implementation has been particularly useful in this instance in the pursuit to achieve this objective.

Technical - There is an opportunity to increase the frequency of electrolyte temperature data, facilitating improved process monitoring and control. Due to the high labour requirement associated with the manual electrolyte temperature measurement technique, bath temperature measurement was previously limited to one measurement per 48.0 hour period. However, labour requirement is not a constraint on the neural network prediction technique, hence, there is an opportunity to increase the frequency of data acquisition. The implications of this are improved electrolyte scheduling and reduction cell control, attributed to more frequent up-to-date data. Hence, neural network implementation for electrolyte temperature prediction has substantial technical merit in addition to the noted economic and ergonomic benefits.

8.5 CONCLUDING REMARKS

The high degree of accuracy exhibited by each of the selected neural networks applied to each of the studied industrial applications demonstrates the ability of each model to interpolate from the training examples used. In each instance, the applied neural networks have shown good capacity to generalise to make accurate predictions for previously unseen data patterns. This result indicates the sufficiency of the training data patterns used in the first instance. Specifically, the data patterns used for

neural network training were sufficient to allow good generalisation by each model when implemented and encountering unknown input vectors. In particualr, it has been shown for each of the neural network models selected for each of the studied industrial applications that the applied model in each instance is capable of achieving high prediction accuracy. Moreover, while this high accuracy is achieved, it is noted that the most economically beneficial neural network is applied in each instance. In particular, it is shown that the GRNN model applied for the electrolyte additive prediction application yields reduced electrolyte temperature variation, compared to the existing additive scheduling technique, while using the minimum number of process parameters as predictors. In addition, a mean electrolyte temperature of approximately 965.0°C is achieved. Hence, the neural network model for the electrolyte additive prediction application achieves the specified objective. Considering the cell failure prediction application, while the WH neural network has been highlighted as the most economically beneficial model to use for the application, while not achieving minimum error of the applied models, it is shown that a high degree of accuracy is obtained. In particular, it is shown that the developed WH model achieved 99.6% accuracy when applied for predicting cell failure. Further, the BP2 neural network has shown relatively high accuracy for predicting electrolyte temperature. It is shown that the prediction error associated with the BP2 model during the implementation investigation is comparable to that achieved for the test data set. Moreover, it is noted that the average prediction error associated with the majority of neural network predictions is comparable to that associated with the manual measurement technique. In each instance, it is noted that neural network implementation has directly resulted in specific economic, ergonomic and technical improvements at CABBL. Hence, the research findings have been directly implemented to achieve particular process improvements in the aluminium smelting process.

CHAPTER NINE

Final Concluding Remarks and Recommendations for Future Work

9.1 FINAL CONCLUDING REMARKS

It is noted that continuous process improvement is a primary focus of many manufacturing industries throughout the world as a means to maintain and increase their competitive level. Moreover, it is inferred that process control is an essential area of operations management in a continuous manufacturing operation, such as aluminium smelting. As a means of process control during aluminium production, it is highlighted that operators are responsible for evaluating the interactions between process parameters and administering appropriate action to maintain or return the overall system to a predetermined control range, bounded by lower and upper control limits. It is noted that the development of a model to identify process relationships and predict parameter values in the continuous aluminium smelting process is complicated due to the nature of the process. Further, parameters of the Hall-Heroult process typically behave dynamically and are functions of complex non-linear relationships and interactions. While it is noted that manufacturing processes typically require a highly adaptive and accurate modelling technique to facilitate the development of a successful model, it is documented that artificial neural networks are a particular modelling technique that are well suited to the industry environment at Comalco Aluminium (Bell Bay) Limited, or CABBL, a major industry in Tasmania, Australia. While it is shown that traditional regression techniques are inadequate at CABBL for the applications studied, it is shown that neural networks offer a successful process modelling and estimation technique. This is attributed to the difficulty of finding such an algorithm capable of performing in the complex, noisy, non-linear, dynamic environment of an aluminium smelting industry. The application of neural networks at CABBL involved translating the mathematical algorithms associated with each of the studied neural network models into

appropriate computer program code to exploit the high throughput of their parallel processing ability. However, the success of neural networks in the applications studied as part of this work is typically due to their ability to generalise from the training data provided. A fundamental requirement for the accuracy of neural network generalisation is that 'from causes that appear similar, we expect similar effects' [237].

While each of the applied neural networks where selected in the first instance because of the suitability of their architecture and associated algorithms, a sensitivity analysis and numerical investigation confirmed the suitability of the selected models. In particular, it was shown that each of the selected models exhibited high performance when exposed to a non-linear, complex and discontinuous mathematical function, that could well have been a simulation of some manufacturing process. Moreover, the predictive and casual importance techniques introduced and applied in this work as a part of the sensitivity analysis demonstrate a useful methodology for removing input nodes and their connections. Further, it is shown that the remaining input nodes and their associated weight connections are adjusted so that the overall input-output behaviour of the neural network remains approximately unchanged. Moreover, in some instances, removal of non-contributing input nodes from the neural network is shown to improve the generalisation ability of the model. In each instance, the ultimate objective of the predictive and casual importance analyses was to reduce an initial set of available features associated with an application to a subset that captured the fundamental information content of the data. While it is noted that the predictive and casual importance techniques applied in this instance where available in the literature, the methodology developed to calculate the percentage contribution of the input parameters in a neural network model was the authors original idea. The technique described was developed as a result of the requirement to be able to rank the significance of a set of parameters in a neural network model. This information is particularly useful to the aluminium industry to further an understanding of the Hall-Heroult process. Nevertheless, it is shown that the ranking of the input parameters using the percentage contribution technique is a precise mathematical procedure that is useful for identifying and ranking the significance of variables in a process model. It is also important to note that the mode of investigation applied in the sensitivity analysis was specifically developed to

establish a foundation for selecting the optimum neural network model, using specific selection criteria, for a given application. Hence, the procedure adopted for identification of an optimum neural network architecture and algorithm, in the first instance, with subsequent establishment of the minimum number of input parameters required in the model and the ranking of importance of the input variables was established to facilitate ultimate selection of an optimum neural network model for a specific application.

The selected neural networks and sensitivity analysis techniques were applied to industrial data from an aluminium smelting process for three specific applications and their suitability assessed from their performance on a common test data set for each application. Specifically, prediction accuracy, number of process parameters required in the model and computation time was documented for each of the applied neural networks for each of the studied applications. For each application, the identification problem involved establishing a suitable mapping between two sets of variables, namely, an input and output data set. Hence, the task of the neural network in each instance was one of function approximation. This study has shown that it is possible to predict particular performance features of the Hall-Heroult process given more easily measured parameters, at least to a reasonable degree of accuracy. In each instance, a statistical analysis has shown that the error associated with each neural network was statistically significantly different to the error associated with any other neural network for each specific application. Further, the features of the training data used by each of the neural networks was different in each instance. Hence, it is shown that when a neural network is initially trained for a particular task some of the features of the training data have no significant effect on the networks decision, while other features are critical. In addition, it is shown that while there are many networks for a particular task that may perform similarly, they each use different features of the training data to make their decision. In addition to accuracy and features of the training data required by each neural network, computation time was shown to be significantly different for each of the applied architectures and algorithms. It has been shown that computation time varies significantly with model complexity and input vector dimensionality. In order to select a particular neural network for each of the studied applications it is shown that specific selection criteria are required to make a quantitative decision. Specifically, neural network error, number of process

parameters required in the model and computation time were considered necessary and sufficient selection criteria. However, it is shown that the decision of which neural network to use for a specific application is complex as the selection criteria are not of equal significance and further, there is no single model that is ranked first for the selection criteria in each instance. Hence, it was necessary to select a neural network using quantitative decision logic. In the first instance, traditional optimisation strategies were investigated, specifically, operations research techniques such as simplex method and integer programming. However, such methodologies were shown to be inappropriate for the optimisation problem considered in this instance. Hence, an efficient and effective optimisation technique was developed. It is shown that the developed optimisation technique is capable of optimising a neural network rapidly on command. In particular, the optimisation strategy performs a quantitative analysis of a neural network model to determine whether the removal of certain process parameters from the neural network yields improved economic benefit from the initial feasible solution. The optimisation strategy developed is particularly useful as it incorporates procedures required for pre-processing of data prior to neural network training and testing. In addition, the optimisation program is written such that any neural network model can be optimised for any given application. Moreover, while it is highlighted that the developed program is a unique and novel optimisation methodology, it is important to note that it is a tailor made tool that has been carefully developed to facilitate its ease of application to industry.

Implementation of the optimum neural network for each of the studied applications has shown that the neural network performance exhibited in the test data set for each application was replicated subsequent to implementation, highlighting the suitability of the test data set developed in each instance and the generalisation capability of the applied models. Moreover, it is shown that neural network implementation in each instance directly resulted in specific economic, ergonomic and technical improvements at the smelter. It is shown that while the neural network modelling completed in this instance yielded economic benefit for electrolyte additive, cell failure and electrolyte temperature prediction, particular ergonomic improvements and technical benefits also emanated from the study. Given the high accuracy of the neural network modelling completed for each of the studied applications, it follows that the majority of the potential savings highlighted for each application are

achievable as a result of neural network implementation. Moreover, the neural network optimisation strategy developed further ensures that the maximum percentage of the potential savings is achieved.

It is useful to note that rather than individually assessing models for their suitability for a given application, this work incorporates strategies to compare many neural network models via quantitative and qualitative comparison methodologies. While it is shown that model error, process parameters required and computation time are used for model assessment, the selection criteria are selected based on the needs of industry. Specifically, the selection criteria are necessary and sufficient to select a neural network model for a specific application based on economic consideration. Moreover, the selection criteria used ensure the neural network selected for the application yields maximum economic benefit. In addition, while the procedure developed and used throughout this investigation for ultimate neural network optimisation and selection is shown to be appropriate for the studied applications, it is important to note that the methodology is applicable to any application. That is, while many neural network models can be assessed and compared for any application, one particular model will be highlighted as a result of the optimisation analysis as most suitable for the application, in regards to economic benefit.

The objectives of the research work completed in this instance are specifically formulated to satisfy the needs of industry. In particular, while the major objective of the research work was to select an optimum neural network model for a specific application, the strategies used to achieve this objective were developed to cater for industry requirements. Specifically, the developed strategies were required to be generic, adaptable and easily implemented. However, while the work completed to date has shown neural network modelling to be extremely useful at CABBL for process monitoring and control, it is necessary to note that the procedure adopted for this study is applicable at other Comalco sites and associated operations. Specifically, the neural network algorithms, optimisation program and investigation and implementation methodology developed and applied as a part of this research work are generic in nature and therefore transportable and adaptable. Hence, the modelling technique applied in this instance at CABBL can be applied at other Comalco

industries and associated operations to maximise the profitability of Comalco and improve the monitoring and control of the aluminium smelting process.

9.2 RECOMMENDATIONS FOR FUTURE WORK

The recommendations for future work documented here include suggestions to further develop the accompanying neural network analysis and optimisation program and moreover, further the application of neural networks at CABBL for process monitoring and control.

Hence, a recommendation for future work is to further develop the accompanying neural network analysis and optimisation program. The accompanying software was developed in the first instance to facilitate numerous pre-processing actions required for neural network training and testing, such as normalising numerical data and creating train and test data sets. In addition, neural network optimisation is incorporated into the program to facilitate the selection of an optimum neural network model for a specific application. However, while the developed program incorporates algorithms to assist complex decisions, there is no option to automatically select an optimum architecture for a given train and test data set. Specifically, it is necessary to determine heuristically the minimum number of hidden layer nodes, for example, to use in a backpropagation neural network. However, this decision could be automated by further developing the accompanying software. This would require extensive programming to incorporate specific algorithms to select a neural network architecture to minimise model complexity and yield maximum performance. It would be useful to translate the neural network programs from Pascal code to Microsoft® Visual Basic™ code to allow communication between the neural network models and the optimisation program. Ultimately, a user with no prior knowledge should be able to enter a particular batch of data representing the external behaviour of some process and then be guided to the selection of an optimal model to implement for an application without the requirement to manually select the optimum architecture to use for each neural network applied. This would eliminate a time consuming and arduous task associated with neural network modelling and conceptually, appears feasible and achievable.

The recommendations that follow are in regard to the application of neural networks at CABBL for process monitoring and control. Moreover, while one recommendation is to identify further areas of the smelting process that would benefit from phenomenological modelling of this nature, it is also noted that the studied applications may be further investigated. In particular, the addition of electrolyte additives to reduction cells is considered. It is noted that the availability of electrolyte temperature data using the manual measurement technique was limited to one measurement per cell per 48.0 hours. However, using the neural network for prediction and estimation of electrolyte temperature yields a technique that can provide more frequent data for this important process parameter. This has significant implications, particularly, the more frequent data may be used to more accurately schedule electrolyte additives to the reduction cell. While it has been shown that electrolyte temperature is an important process parameter to consider for the addition of AlF_3 and Na_2CO_3 to the reduction cell, the existing technique may incorporate an electrolyte temperature that was measured up to a maximum of 48.0 hours prior, hence, the data may be inaccurate. However, more frequent data may improve the prediction of AlF_3 and Na_2CO_3 and ultimately improve reduction cell control and therefore is worth further investigation.

In addition, the control of the aluminium reduction cell may be improved through the development of a closed loop process for electrolyte additive additions. In particular, replacing manual additions of AlF_3 and Na_2CO_3 with automated procedures, such as storage hoppers located on the reduction cell. In this instance, neural networks can be utilised to make lower quantity predictions of AlF_3 and Na_2CO_3 with more frequent additions, rather than one large addition per 24.0 hour period. This may yield lower process variation and therefore, improved process efficiency. A current limitation of the existing technique is the requirement for manual additions, limiting the resources available for frequent electrolyte additive additions.

In regard to cell failure prediction, while the developed model is useful for identifying reduction cells that are at high risk of failure, it would be useful to develop a prediction model that was capable of identifying the mode of cell failure prior to the occurrence. While various cell failure mechanisms, such as sidewall tap-out and cathode failure, for example, have been noted, it is useful to highlight that

preventative action may be applied to a high failure risk reduction cell to prolong cell life if the failure mode is known. For instance, if it is predicted that a pothole failure will occur, particular action can be taken to identify the affected cathode section and subsequently, reduce energy input to that section of the cathode. This action significantly reduces the probability of a tap-out occurrence and has an economic benefit in that cell life is extended.

A final recommendation for future work is to investigate further applications of neural networks in the aluminium smelting industry. There are many process parameters associated with the production of aluminium, including environmental considerations, Hall-Heroult process and metal solidification. However, while there are many hard to measure parameters in the production facility, there are typically many other parameters that are correlated with those that are hard to measure. Hence, empirical modelling is applicable, particularly neural network modelling, as it has proven ability in the highly dynamic, unstable, complex environment of the aluminium smelting industry.

References

[1]. GILBERT, A., Keynote Speech, Plenary Session, Conf. Eng. Vision (enVision), Exhibition Centre, Melbourne, Australia, Oct. 26, 1998.

[2]. AMERICAN SOCIETY FOR METALS, "Aluminium: Design and Application", Edited by K. R. Van Horn, vol. 2, American Society for Metals, 1967.

[3]. BOYER, H. E. and HALL, T. L., "Metals Handbook", Desk Edition, American Society for Metals, 1985.

[4]. GOVETT, M. H. and LARSEN, J. E., "Aluminium: The Next Phase", vol. 2, Australian Mineral Economics Limited, 1988.

[5]. TOMAGO ALUMINIUM, "The Aluminium Industry in Australia", http://www.tomago.com.au/company.html, Feb. 24, 1999.

[6]. SOCIETY OF AUTOMOTIVE ENGINEERS, "Aluminium Road Wheels", AFS Transactions, vol. 79, 1986, pp. 1-38.

[7]. HAUPIN, W., "Bath Properties and How They Affect Cell Operation", Proc. 15th International Course on Process Metallurgy of Aluminium, Trondheim, Norway, Jun. 03-07, 1996, pp. 9.1 - 9.33.

[8]. GORDON, V., "The Alchemical Twins", Review, no. 47, Rio Tinto plc, London, Sep. 18, 1998, pp. 15-19.

[9]. STILL, R. F., "Performance Drivers of the Smelting Process", Proc. Aluminium Smelting Fundamentals, Course 1, Comalco Aluminium Limited, 1994.

[10]. AMSTEAD, B. H., OSTWALD, P. F. and BEGEMAN, M. L., "Manufacturing Processes", Eighth Edition, McGraw-Hill, 1987.

[11]. CRISP, A. J., BROWN, G. P., RODDA, D. P. and GOODES, C. G., "Alumina Production: Principles and Practice", Proc. Aluminium Smelting Fundamentals, Course 1, Comalco Aluminium Limited, 1994.

[12]. WORLD ALUMINIUM, "Manufacturing Process: Alumina", http://www.world-aluminium.org/media/bottom_main.jpg, Jul. 03, 2000.

[13]. GRJOTHEIM, K. and WELCH, B. J., "Aluminium Smelter Technology", Second Edition, Aluminium-Verlag, Dusseldorf, 1988.

[14]. LOGAN, R. H., "What is Electrolysis?", http://edie.cprost.stu.ca/~rhlogan/electrol.html, Oct. 13, 1998.

[15]. HAUPIN, W. E., "Principles of Aluminium Electrolysis", Proc. 124th TMS Annual Meeting, Las Vegas, Feb. 12-16, 1995, pp. 195-203.

[16]. TOMAGO ALUMINIUM, "The Aluminium Production Process", http://www.tomago.com.au/aluminium.html, Feb. 24, 1999.

[17]. LIU, X., "Role and Performance of Electrolyte in Aluminium Smelting", Proc. Aluminium Smelting Fundamentals, Course 2, Comalco Aluminium Limited, 1994.

[18]. WORLD ALUMINIUM, "Smelter Technology", http://www.world-aluminium.org/media/tech2.gif, Jul. 03, 2000.

[19]. KENIRY, J., "Outline of the Reduction Process", Proc. Aluminium Smelting Fundamentals, Course 1, Comalco Aluminium Limited, 1994.

[20]. GRJOTHEIM, K. and KVANDE, H., "Understanding the Hall-Heroult Process for Production of Aluminium", Aluminium-Verlag, Dusseldorf, 1986.

[21]. HUME, S., "Anode Raw Materials", Proc. Aluminium Smelting Fundamentals, Course 3, Comalco Aluminium Limited, 1994.

[22]. SADLER, B. A. and WELCH, B. J., "Anode Consumption and the Ideal Anode Properties", Proc. Aluminium Smelting Fundamentals, Course 3, Comalco Aluminium Limited, 1994.

[23]. TAYLOR, M. P., ZHANG, W. D, WILLIS, V. and SCHMID, S., "A Dynamic Model for the Energy Balance of an Electrolysis Cell", Trans. Institute of Chemical Engineers, vol. 74, pt. A, Nov. 1996, pp. 913-933.

[24]. TABEREAUX, A. T., ALCORN, T. R. and TREMBLEY, L., "Lithium-Modified Low Ratio Electrolyte Chemistry for Improved Performance in Modern Reduction Cells", Proc. 122nd TMS Annual Meeting, Denver, Colorado, Feb. 21-25, 1993, p. 221.

[25]. UTIGARD, T. A., "Density of the Na_3AlF_6-AlF_3-Al_2O_3-CaF_2 System: A Key to the Performance of Hall-Heroult Cells", Proc. 122nd TMS Annual Meeting, Denver, Colorado, Feb. 21-25, 1993, pp. 239-245.

[26]. GRJOTHEIM, K., KVANDE, H. and WELCH, B. J., "Low-Melting Baths in Aluminium Electrolysis", Proc. 115th TMS Annual Meeting, New Orleans, Louisiana, Mar. 02-06, 1986, pp. 417-423.

[27]. MATHEOU, N., "Electrolyte Composition Control in an Aluminium Reduction Cell", Proc. Aluminium Smelting Fundamentals, Course 1, Comalco Aluminium Limited, 1994.

[28]. TARCY, G. P. and SORENSEN, J., "Determination of Factors Affecting Current Efficiency in Commercial Hall Cells Using Controlled Potential Coulometry and Statistical Experiments", Proc. 120th TMS Annual Meeting, New Orleans, Louisiana, Feb. 17-21, 1991, pp. 453-459.

[29]. TAYLOR, M. P., "Static and Dynamic Energy Balance in Reduction Cells", Proc. International Course on Process Metallurgy of Aluminium, Trondheim, Norway, May 24-28, 1993.

[30]. SOLHEIM, A., "Liquidus Temperature and Alumina Solubility in the System Na_3AlF_6-AlF_3-LiF-CaF_2-MgF_2", Proc. 124th TMS Annual Meeting, Las Vegas, Feb. 12-16, 1995, pp.451-460.

[31]. DEWING, E. W., "Loss of Current Efficiency in Aluminium Electrolysis Cells", Met. Trans. B, vol. 22B, 1991, pp. 177-182.

[32]. KUSCHEL, G. I. and WELCH, B. J., "Further Studies of Alumina Dissolution Under Conditions Similar to Cell Operation", Proc. 120th TMS Annual Meeting, New Orleans, Louisiana, Feb. 17-21, 1991, pp. 299-305.

[33]. SEGATZ, M. and DROSTE, C., "Analysis of Magnetohydrodynamic Instabilities in Aluminium Reduction Cells", Proc. 123rd TMS Annual Meeting, San Francisco, Feb. 27 - Mar. 03, 1994, p. 413.

[34]. URATA, N., "Magnetics and Metal Pad Instability", Proc. 114th TMS Annual Meeting, Atlanta, Georgia, Mar. 06-10, 1985, pp. 581-591.

[35]. FISHER, W. K., "Interdependence Between Anode Net Consumption and Pot Design, Pot Operating Parameters and Anode Properties", Proc. 120th TMS Annual Meeting, New Orleans, Louisiana, Feb. 17-21, 1991, pp. 681-686.

[36]. SORLIE, M., HVISTENDAHL, J. and OYE, H. A., "Early Failure Mechanisms in Aluminium Cell Cathodes", Proc. 122nd TMS Annual Meeting, Denver, Colorado, Feb. 21-25, 1993, pp. 299-308.

[37]. VERSTREKEN, P., "Bath and Liquidus Temperature Sensor for Molten Salts", Proc. 125th TMS Annual Meeting, Anaheim, California, Feb. 04-08, 1996, pp. 437-444.

[38]. TAYLOR, M. P., "Challenges in Optimising and Controlling the Electrolyte in Aluminium Smelters", Proc. International Conference on Molten Slags, Fluxes and Salts, Sydney, Australia, 1997, pp. 659-674.

[39]. SALT, D. J., "Bath Chemistry Control System", Proc. 119th TMS Annual Meeting, Anaheim, California, Feb. 18-22, 1990, pp. 299-304.

[40]. OPHARDT, C., "Conversion of Bauxite Ore to Aluminium Metal: Electrolysis Cell - Hall Process", http://elmhcx9.elmhurst.edu/~chm/onlcourse/chm110/outlines/aluminum.html, Oct. 13, 1998.

[41]. KEELER, J. D., "Prediction and Control of Chaotic Chemical Reactions via Neural Network Model", Proc. Conference on Artificial Intelligence in Petroleum Exploration and Production (CAIPEP), Plano, TX, 1993, pp. 1-7.

[42]. UNGER, L. H., ERIC, J. H., KEELER, J. D. and MARTIN, G. D., "Process Modelling and Control Using Neural Networks", Proc. Intelligent Systems in Process Engineering (ISPE), Snowmass, CO, 1995, pp. 1-11.

[43]. CAUDILL, M. and BUTLER, C., "Naturally Intelligent Systems", Massachusetts Institute of Technology, 1990.

[44]. ZURADA, J. M., "Introduction to Artificial Neural Systems", West Publishing Company, 1992.

[45]. McCULLOCH, W. S. and PITTS, W., "A Logical Calculus of the Ideas Immanent in Nervous Activity", Bulletin of Mathematical Biophysics, vol. 5, 1943, pp. 115-133.

[46]. HERB, D. O., "The Organisation of Behaviour", Wiley, New York, 1949.

[47]. ROSENBLATT, F., "The Perceptron: A Probabilistic Model for Information Storage and Organisation in the Brain", Psychological Review, vol. 65, 1960, pp. 386-408.

[48]. HERTZ, J., KROGH, A. and PALMER, R. G., "Introduction to the Theory of Neural Computing", Addison-Wesley Publishing Company, 1991.

[49]. WIDROW, B. and HOFF, M. E., "Adaptive Switching Circuits", 1960 IRE WESCON Convention Record, pt. 4, New York, 1960, pp. 96-104.

[50]. MINSKY, M. and PAPERT, S., "Perceptrons: An Introduction to Computational Geometry", M. I. T. Press, 1969.
[51]. KOHONEN, T., "Associative Memory: A System-Theoretical Approach", Berlin, Springer-Verlag, 1977.
[52]. ANDERSON, J. and ROSENFELD, E., "Neurocomputing: Foundations of Research", M. I. T. Press, Cambridge, MA, 1988.
[53]. HOPFIELD, J. J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", Proc. Natl. Acad. Sci., vol. 79, 1982, pp. 2554-2558.
[54]. HOPFIELD, J. J., "Neurons With Graded Response Have Collective Computational Properties Like Those of Two State Neurons", Proc. Natl. Acad. Sci., vol. 81, 1984, pp. 3088-3092.
[55]. RUMELHART, D. E. and McCLELLAND, J. L., "Parallel Distributed Processing", vol. 2, M.I.T. Press, Cambridge, MA, 1986.
[56]. RUMELHART, D. E. and McCLELLAND, J. L., "Parallel Distributed Processing: Explorations in the Microstructure of Cognition, I and II", M. I. T. Press, Cambridge, MA, 1986.
[57]. HECHT-NIELSEN, R., "Neurocomputing", HNC, Inc., 1989.
[58]. ROJAS, R., "Neural Networks: A Systematic Introduction", Springer-Verlag Berlin Heidelberg, 1996.
[59]. CAUDILL, M. and BUTLER, C., "Understanding Neural Networks - Computer Explorations", vol. 1, Massachusetts Institute of Technology, 1992.
[60]. WEIJTERS, A. J. M. M. and HOPPENBROUWES, G. A. J., "Backpropagation Networks for Grapheme-Phoneme Conversion: A Non-Technical Introduction", Artificial Neural Networks, Springer-Verlag Berlin Heidelberg, 1995, pp. 13-36.
[61]. DAVALO, E. and NAIM, P., "Neural Networks", The MacMillan Press Limited, 1991.
[62]. FREEMAN, J. A. and SKAPURA, D. M., "Neural Networks: Algorithms, Applications and Programming Techniques", Addison-Wesley Publishing Company, Inc., 1992.
[63]. MASTERS, T., "Practical Neural Network Recipes in C++", Academic Press Inc., 1993.
[64]. CHOW, M., SHARPE, R. N. and HUNG, J. C., "On the Application and Design of Artificial Neural Networks for Motor Fault Detection - Part I", IEEE Transactions on Industrial Electronics, vol. 40, no. 2, 1993, pp.181-188.
[65]. VEELENTURF, L. P. J., "Analysis and Applications of Artificial Neural Networks", Prentice Hall International (UK) Limited, 1995.
[66]. GOLDEN, R. M., Mathematical Methods for Neural Network Analysis and Design", Massachusetts Institute of Technology, 1996.
[67]. KARTALOPOULOS, S. V., "Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications", IEEE, Inc., New York, 1996.
[68]. FAUSETT, L., "Fundamentals of Neural Networks: Architectures, Algorithms and Applications", Prentice Hall International, Inc., 1994.
[69]. BATTELLE, "Neural Networks: What are Neural Networks?", http://www.emsl.pnl.gov: 2080/proj/neuron/neural/what.html, Jun. 11, 1998.
[70]. RAO, D. H., GUPTA, M. M. and WOOD, H. C., "Neural Networks in Control Systems", Journ. Neural Networks: Theory, Technology and Applications, IEEE, 1996.
[71]. WASSERMAN, P. D., "Advanced Methods in Neural Computing", Van Nostrand Reinhold, 1993.
[72]. HUANG, S. H. and ZHANG, H. C., "Artificial Neural Networks in Manufacturing: Concepts, Applications and Perspectives", IEEE Transactions on Components, Packaging and Manufacturing Technology, part A, vol. 17, no. 2, 1994, pp. 212-228.
[73]. LIPPMAN, R. P., "An Introduction to Computing with Neural Nets", IEEE ASSP Mag., Apr. 24, 1987.
[74]. WILLIS, M. J., MONTAGUE, G. A. and PEEL, C., "On the Application of Artificial Neural Networks for Process Control", Kluwer Academic Publishers, London, 1995.
[75]. HAYKIN, S., "Neural Networks: A Comprehensive Foundation", Macmillan College Publishing Company, Inc., 1994.
[76]. CHEN, C. H., "Fuzzy Logic and Neural Network Handbook", McGraw-Hill Companies, Inc., 1996.
[77]. KANDIL, N., KHORASANI, K., PATEL, R. V. and SOOD, V. K., "Optimum Learning Rate for Backpropagation Neural Networks", Neural Networks: Theory, Technology and Applications, IEEE, 1996.
[78]. WATSON, M., "Common LISP Modules: Artificial Intelligence in the Era of Neural Networks and Chaos Theory", Springer-Verlag New York, Inc., 1991.

[79]. ZEIDENBERG, M., "Neural Networks in Artificial Intelligence", New York, Ellis Horwood, 1990.
[80]. LUNDSTEDT, H., "What is a Neural Network?", http://nastol.astro.lu.se/~henrik/neuramet2.html, Oct. 13, 1998.
[81]. DAYHOFF, J. E., "Neural Network Architectures - An Introduction", Van Nostrand Reinhold, 1990.
[82]. MEHROTRA, K. G., MOHAN, C. K. and RANKA, S., "Bounds on the Number of Samples Needed for Neural Learning", IEEE Transactions on Neural Networks, vol. 2, no. 6, Nov., 1991, pp. 548-558.
[83]. YU. X., LOH, N. K., JULLIEN, G. A. and MILLER, W. C., "Comparisons of Four Learning Algorithms for Training the Multilayer Feed Forward Neural Networks with Hard Limiting Neurons", Neural Networks Theory, IEEE, New York, 1996.
[84]. BADIRU, A, B., "Expert System Applications in Engineering and Manufacturing", Englewood Cliffs, NJ, Prentice-Hall, 1992.
[85]. UDO, G. J., "Neural Network Applications in Manufacturing Processes", Computers and Industrial Engineering", vol. 23, no. 1-4, 1992, pp. 97-100.
[86]. CARDON, H. R. A and HOOGSTRATEN, R, V, "Key Issues for Successful Industrial Neural Network Applications: An Application in Geology", Artificial Neural Networks, Springer-Verlag Berlin Heidelberg, 1995, pp. 235-245.
[87]. ELSIMARY, H., MASHALI, S. and SHAHEEN, S., "A Method for Training Feed Forward Neural Networks to be Fault Tolerant", Neural Networks Theory, IEEE, Inc., New York, 1996.
[88]. MAREN, A., HARSTON, C. and PAP, R., "Handbook of Neural Computing Applications", Academic Press Inc., UK, 1990.
[89]. WILLIS, M. J., MONTAGUE, G. A. and PEEL, C., "On the Application of Artificial Neural Networks to Process Control", Kluwer Academic Publishers, Boston, 1995, pp. 191-219.
[90]. LEE, S. and PARK, J., "Neural Computation for Collision Free Path Planning", Journ. Intelligent Manufacturing, vol. 2, no. 5, 1991, pp. 15-326.
[91]. CHRYSSOLOURIS, G., DOMROESSE, M. and BEAULLIEU, P., "Sensor Synthesis for Control of Manufacturing Processes", Journ. Engineering for Industry, vol. 114, May, 1992, pp. 158-174.
[92]. SORSA, T, KOIVO, N. and KOIVISTO, H., "Neural Networks in Process Fault Diagnosis", IEEE Trans. Syst., Man. and Cybern., vol. 21, no. 4, 1991, pp. 815-825.
[93]. RAY, A. K., "Equipment Fault Diagnosis: A Neural Network Approach", Computers in Industry, vol. 16, 1991, pp. 169-178.
[94]. WASSERMAN, P. D., UNAL, A. and HADDAD, S., "Neural Networks for On-Line Machine Condition Monitoring", Intelligent Engineering Systems Through Artificial Neural Networks, New York, ASME Press, 1991, pp. 693-700.
[95]. KOTANI, M., UEDA, Y., MATSUMOTO, H. and KANAGAWA, T., "Acoustic Diagnosis for Blower with Wavelet Transform and Neural Networks", IEEE, Inc., 1995.
[96]. BURKE, L. I. and RANGWALA, S., "Tool Condition Monitoring in Metal Cutting: A Neural Network Approach", Journ. Intelligent Manufacturing, vol. 2, no. 5, 1991, pp. 269-280.
[97]. ELANAYAR, S. and SHIN, Y. C., "Tool Wear Estimation in Turning Operations Based on Radial Basis Functions", Intelligent Engineering Systems Through Artificial Neural Networks, New York, ASME Press, 1991, pp. 685-692.
[98]. GOVEKAR, E. and PEKLENIK, H., "Monitoring of a Drilling Process by Neural Network", 21st CIRP Int. Seminar on Manuf. Systems, Stockholm, Sweden, Jun. 07-10, 1989.
[99]. KARRI, V., "Performance Estimation in Wood Machining Using ANN", Proc. Advanced Manufacturing Processes, Systems and Technologies (AMPST), Bradford, England, Mar. 21-24, 1996, pp. 271-276.
[100]. ISLAM, M. M., RAHMAN, S. M. and SARKER, R., "A Neural Network Model for Invariant Pattern Recognition", Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), World Scientific, 1998, pp. 318-323.
[101]. ZHIZHAI, H. and ZHANG, M., "Self-Organising Neural Network Based Discrete Optical Flow Model for Face Recognition", Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), World Scientific, 1998, pp. 372-377.
[102]. WIDROW, H. and WINTER, R., "Neural Nets for Adaptive Filtering and Adaptive Pattern Recognition", Computer, vol. 21, Mar. 12, 1988, pp. 25-39.
[103]. SOBAJIC, D. J., LU, J. J. and PAO, Y. H., "Intelligent Control for the Intelledex 605 T Robot Manipulator", Proc. 1998 IEEE Int. Neural Networks Conf., vol. 2, 1988, pp. 613-640.

[104]. ALBUS, J., "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)", Journ. Dynamic Systems, Measurement and Control, Sep. 11, 1975, pp. 220-227.
[105]. TAYLOR, J. G., "The Promise of Neural Networks", Springer-Verlag London Limited, 1993.
[106]. PAN, X. W., "Control Model for Samancor M10 Furnace via Neural Networks, Meyerton Plant Network Report 1, Samancor, Oct. 06, 1996, pp. 1-42.
[107]. PAN, X. W., "Advanced Control of West Plant using Pavillion Neural Networks and G2 Expert System", Meyerton Plant Network Report 2, Samancor, Nov. 07, 1996, pp. 1-19.
[108]. SPANGLER, M. V., "Use of Process Insights at a Refractory Gold Plant, Proc. Pavillion Users Conference, Oct. 17, 1994, pp. 1-7.
[109]. TATSURO, H., KOZO, Y., SHINOBU, M. and HIROSHI, T., "Blast Furnace Operation System Using Neural Networks and Knowledge Base", Proc. Sixth International Iron and Steel Congress, Nagoya, ISIJ, 1990, pp. 23-27.
[110]. SWINGLER, K., "Applying Neural Networks: A Practical Guide", Academic Press Limited, 1996.
[111]. GINGRICH, C. G., "Modelling Human Operators Using Neural Networks", ISA Transactions, vol. 31, no. 3, 1992, pp. 81-90.
[112]. JALEL, N. A., MIRZAI, A. R., LEIGH, J. R. and NICHOLSON, H., "Applications of Neural Networks in Process Control", Neural Network Applications, London, Springer-Verlag, 1991, pp. 101-113.
[113]. CHRYSSOLOURIS, G. and GUILLOT, M., "An AI Approach to the Selection of Process Parameters in Intelligent Machining", ASME on Sensors and Controls for Manufacturing, WAM, Chicago, IL, Nov. 08, 1988.
[114]. MADEY, G. R., WEINROTH, J. and SHAH, V., "Integration of Neurocomputing and System Simulation for Modelling Continuous Improvement Systems in Manufacturing", Journ. Intelligent Manufacturing, vol. 3, 1992, pp. 193-204.
[115]. TRIPPI, R. R. and TURBAN, E., "Neural Networks in Finance and Investing", Proc. Advances in Neural Information Processing, 1993, pp. 211-219.
[116]. BRUNELLI, R., "Training Neural Nets Through Stochastic Minimisation", Journ. Neural Networks, vol. 7, no. 9, 1994, pp. 1405-1412.
[117]. MAMMONE, R. J. and ZEEVI, Y., "Neural Networks: Theory and Applications", Academic Press, Inc., 1991.
[118]. RUMELHART, D. E. and McCLELLAND, J. L., "Parallel Distributed Processing", vol.1, A Bradford Book, 1988.
[119]. KARRI, V., "Practical Application of Neural Nets for Manufacturing Processes", Proc. Evolutionary Models in Engineering Conference (EMEC), Auckland, Sep. 01-04, 1995, pp.103-113.
[120]. TARNG, Y. S., HSEIB, Y. W. and HWANG, D., "Sensing Tool Breakage with Neural Network", Int. Mach. Tools Manuf., vol. 34, no. 3, 1994, pp.341-350.
[121]. BARSCHDORFF, D. and FEMMER, U., "Artificial Neural Networks for Wear Estimation", Int. Conf. on Int. Manuf. Sys., Vienna, Jun. 13-15,1994, pp.157-161.
[122]. BARSCHDORFF, D. and MONOSTORI, L., "Neural Networks: Their Application and Perspectives in Intelligent Machining", First CIRP Workshop of the Intelligent Manufacturing Systems Seminars on Learning in IMS, Elsevier Science Pub., Budapest, Hungary, Jul. 06-08, 1992, pp.101-119.
[123]. KARRI, V. and FROST, F., "An Intelligent System for Detection of Failed Aluminium Wheels", Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), Gold Coast, Australia, Feb. 10-12, 1997, pp. 147-151.
[124]. BROOMHEAD, D. S. and LOWE, D., "Multi-Variable Functional Interpolation and Adaptive Networks", Complex Systems 2, 1988, pp. 321-355.
[125]. MOODY, J. E. and DARKEN, C. J., "Fast Learning in Networks of Locally Tuned Processing Units", Neural Computation 1, 1989, pp. 281-294.
[126]. RENALS, S., "Radial Basis Function Network for Speech Pattern Classification", Electronics Letters 25, 1989, pp. 437-439.
[127]. POGGIO, T. and GIROSI, F., "Networks for Approximating and Learning", Proc. IEEE, vol. 78, no. 9, 1990, pp. 1481-1497.
[128]. ORR, M. J. L., "Radial Functions", http://www.cns.edu.ac.uk/people/mark/intro/node7.html, Oct. 19, 1998.
[129]. GIROSI, F. and POGGIO, T., "Networks and the Best Approximation Property", Biological Cybernetics, vol. 63, 1990, pp. 169-176.

[130]. LOWE, D., "Radial Basis Function Networks", Neural Computing Research Group, Aston University, Aston Triangle, Birmingham, 1988, pp. 1-14.
[131]. LUNDSTEDT, H., "Radial Basis Function Network", http://nastol.astro.lu.se/~henrik/rbf.html, Oct. 13, 1998.
[132]. LOWE, D., "Radial Basis Function Networks and Statistics", Neural Computing Research Group, Aston University, Aston Triangle, Birmingham, 1988, pp. 1-32.
[133]. SONG, X. M., "Radial Basis Function Networks", http://www.cs.helsinki.fi/~xianming/thesis/m_conten.html, Oct. 13, 1998.
[134]. SAHA, A., CHRISTIAN, J., TANG, D. S. and Wu, C. L., "Oriented Non-Radial Basis Functions for Image Coding and Analysis", Advances in Neural Information Processing Systems 3, San Mateo, CA, Morgan Kaufmann, 1991, pp. 728-734.
[135]. POGGIO, T. and EDELMAN, S., "A Network That Learns to Recognise Three Dimensional Objects", Nature (London), vol. 343, 1990, pp. 263-266.
[136]. NG, K. and LIPPMANN, R. P., "Practical Characteristics of Neural Network and Conventional Pattern Classifiers", Advances in Neural Information Processing Systems 3, San Mateo, CA, Morgan Kaufmann, 1991, pp. 970-976.
[137]. NIRANJAN, M. and FALLSIDE, F., "Neural Networks and Radial Basis Functions in Classifying Static Speech Patterns", Computer Speech and Language, vol. 4, 1990, pp. 275-289.
[138]. HE, X. and LAPEDES, A., "Nonlinear Modelling and Prediction by Successive Approximation Using Radial Basis Functions", Technical Report LA-UR-91-1375, Los Alamos National Library, Los Alamos, NM, 1991.
[139]. KADIRKAMANATHAN, V., NIRANJAN, M. and FALLSIDE, F., "Sequential Adaptation of Radial Basis Function Neural Networks", Advances in Neural Information Processing Systems 3, San Mateo, CA, Morgan Kaufmann, 1991, pp. 721-727.
[140]. LOWE, D. and WEBB, A. R., "Exploiting Prior Knowledge in Network Optimisation: An Illustration from Medical Prognosis", Network 1, 1990, pp. 299-323.
[141]. KOHONEN, T., "Self-Organisation and Associative Memory", Berlin, Springer-Verlag, 1984.
[142]. KOHONEN, T., "Adaptive, Associative and Self-Organisation Functions in Neural Computing", Applied Optics, vol. 26, 1987, pp. 4910-4918.
[143]. KOHONEN, T., "Self-Organised Formation of Topologically Correct Feature Maps", Biological Cybernetics, vol. 43, 1982, pp. 59-69.
[144]. KOHONEN, T., "An Introduction to Neural Computing", Neural Networks, vol. 1, 1988, p. 4.
[145]. CAUDILL, M. and BUTLER, C., "Understanding Neural Networks - Computer Explorations", vol. 2, Massachusetts Institute of Technology, 1992.
[146]. LEEM, C. S., DORNFIELD, D. A., and DREYFUS, S. E., "A Customised Neural Network for Sensor Fusion in On-Line Monitoring of Cutting Tool Wear", Journ. Engineering for Industry, Trans. ASME, vol. 117, 1995, pp. 152-159.
[147]. SPECHT, D. F., "General Regression Neural Networks", IEEE Transactions on Neural Networks, vol. 2, no. 6, Nov., 1991, pp. 568-576.
[148]. SARLE, W., "What is a GRNN?", ftp://ftp.sas.com/pub/neural/FAQ.html, Nov. 17, 1998.
[149]. CHEN, C. H., "Fuzzy Logic and Neural Network Handbook", The McGraw-Hill Companies, Inc., 1996, pp. 3.1- 3.44.
[150]. SHAFFER, R., "General Regression Neural Networks", http://cheml.nrl.navy/~shatter/grnn.html, Nov. 17, 1998.
[151]. MASTERS, T., "Advanced Algorithms for Neural Networks: A C++ Sourcebook", John Wiley and Sons, 1995.
[152]. NARENDRA, K. S. and PARTHASARATHY, K., "Identification and Control of Dynamical Systems Using Neural Networks", IEEE Transactions on Neural Networks, vol. 1, Mar. 22, 1990, pp. 4-27.
[153]. FREUND, J. E. and SIMON, G. A., "Modern Elementary Statistics", Eighth Edition, Prentice-Hall, Inc., 1992.
[154]. MENDENHALL, W., REINMUTH, J. E. and BEAVER, R., "Statistics for Management and Economics", Sixth Edition, PWS-KENT Publishing Company, 1989.
[155]. MOORE, D. S. and McCABE, G. P., "Introduction to the Practice of Statistics", W. H. Freeman and Company, 1989.
[156]. TROCHIM, W. M., "Hypotheses", http://trochnim.human.cornell.edu/kb/hypotheses.htm, Mar. 29, 2000.
[157]. TROCHIM, W. M., "The t-Test", http://trochnim.human.cornell.edu/kb/power.htm, Mar. 29, 2000.

[158]. MONTGOMERY, D. C., "Introduction to Statistical Quality Control", Sixth Edition, John Wiley and Sons, Inc., 1991.
[159]. LANE, D. M., "Introduction to Tests Supplementing a One-Factor Between-Subjects ANOVA", http://www.ruf.rice.edu/~lane/hyperstat/B93758.html, Mar. 29, 2000.
[160]. LANE, D. M., "Type I and II Errors", http://www.ruf.rice.edu/~lane/hyperstat/A18652.html, Mar. 29, 2000.
[161]. TROCHIM, W. M., "Statistical Power", http://trochnim.human.cornell.edu/kb/power.htm, Mar. 29, 2000.
[162]. SARLE, W., "How to Measure Importance of Inputs", ftp://ftp.sas.com/pub/ neural/FAQ.html, Apr. 24, 1999.
[163]. FROST, F. and KARRI, V., "Determining the Influence of Input Parameters on BP Neural Network Output Error Using Sensitivity Analysis", Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), New Delhi, INDIA, Sep. 23-26, 1999.
[164]. SARLE, W. (saswss@unx.sas.com), "Input Variable Importance", e-mail to Fred Frost (Fred.Frost@comalco.riotinto.com.au), May 11, 1999.
[165]. ANTON, H., "Elementary Linear Algebra", Fifth Edition, Anton Textbooks, Inc., 1987.
[166]. POLIT, D. F., "Data Analysis and Statistics", Appleton and Lange, 1996.
[167]. KENETT, R. S. and ZACKS, S., "Modern Industrial Statistic - Design and Control of Quality and Reliability", Brooks/Cole Publishing Company, 1998.
[168]. LEHMANN, M. and ZEITZ, P., "Statistical Explorations with Microsoft® Excel", Brooks/Cole Publishing Company, 1998.
[169]. HASSOUN, M. H., "Fundamentals of Artificial Neural Networks", Massachusetts Institute of Technology, 1995.
[170]. PHAM, D. T. and LIU. X., "Neural Networks for Identification, Prediction and Control", Springer-Verlag London Limited", 1995.
[171]. KARRI, V. and FROST, F., "Effect of Altering the Gaussian Function Receptive Field Width in RBF Neural Networks on Aluminium Fluoride Prediction in Industrial Reduction Cells", Proc. International Conference on Neural Information Processing (ICONIP), Perth, Australia, Nov. 20-25, 1999, pp. 101–106.
[172]. FROST, F. and KARRI, V., "Performance Comparison of BP and GRNN Models of the Neural Network Paradigm Using a Practical Industrial Application", Proc. International Conference on Neural Information Processing (ICONIP), Perth, Australia, Nov. 20-25, 1999, pp. 1069–1074.
[173]. KARRI, V. and FROST, F., "Combined Kohonen and RBF Networks to Predict Electrolyte Additives in Hall-Heroult Cell", Proc. International Conference on Advances in Intelligent Systems: Theory and Applications (AISTA), Canberra, Australia, Feb. 11-15, 2000, pp. 19-24.
[174]. FROST, F. and KARRI, V., "Intelligent Control of Aluminium Reduction Cells Using Backpropagation Neural Networks", Proc. International Conference on Advances in Intelligent Systems: Theory and Applications (AISTA), Canberra, Australia, Feb. 11-15, 2000, pp. 350-356.
[175]. FROST, F. and KARRI, V., "Identifying Significant Parameters for Hall-Heroult Process Modelling using General Regression Neural Network", Proc. 13th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, (IEA/AIE), New Orleans, USA, Jun. 10-13, 2000, pp. 73-78.
[176]. STEVENS, F. J., ZHANG, W. D., TAYLOR, M. P. and CHEN, J., "The Interaction Between Current Efficiency and Energy Balance in Aluminium Reduction Cells", Proc. 121st TMS Annual Meeting, San Diego, California, Mar. 01-05, 1992, pp. 541-547.
[177]. LIU, X., "CaF_2 Control in Comalco Smelters", Comalco Research Centre, TN, 1993, p. 1329.
[178]. HAUPIN, W., "The Influence of Additives on Hall-Heroult Bath Properties", Journ. Metals, Nov. 08, 1991, pp. 28-34.
[179]. ENTNER, P. M., "Further Developments of the AlF_3 Model", Proc. 122nd TMS Annual Meeting, Denver, Colorado, Feb. 21-25, 1993, pp. 265-268.
[180]. WILSON, M. J., "Practical Considerations Used in the Development of a Method for Calculating Aluminium Fluoride Additions Based on Cell Temperature", Proc. 121st TMS Annual Meeting, San Diego, California, Mar. 01-05, 1992, pp. 375-378.
[181]. WESOLOWSKY, G. O., "Multiple Regression and Analysis of Variance: An Introduction for Computer Users in Management and Economics", John Wiley & Sons, Inc., 1976.
[182]. TABACHNICK, B. G. and FIDELL, L. S., "Using Multivariate Statistics", Second Edition, Harper Collins Publishers, Inc., 1989.
[183]. PRAGER, A., "Bath Chemistry Control System for NZAS", Version 1.1, Oct. 16, 1992.

[184]. CHRENKOVA, M., DANEK, V., SILNY, A. and UTIGARD, T. A., "Density, Electrical Conductivity and Viscosity of Low Melting Baths for Aluminium Electrolysis", Proc. 125th TMS Annual Meeting, Anaheim, California, Feb. 04-08, 1996, pp. 227-232.
[185]. TODD, G., "Bath Resistivity Measurement", HMB Operations Folder, Comalco Aluminium (Bell Bay) Limited, Oct. 11, 1993, pp. 2G1-2G4.
[186]. TODD, G., "EMF", HMB Operations Folder, Comalco Aluminium (Bell Bay) Limited, Oct. 11, 1993, pp. 2E1-2E3.
[187]. FITZGERALD, A. E, HIGGINBOTHAM, D. E. and GRABEL, A., "Basic Electrical Engineering", Fifth Edition, McGraw-Hill Inc., 1981.
[188]. CLELLAND, C. H., KENIRY, J. T. and WELCH, B. J., "A Study of Some Aspects of the Influence of Cell Operation on Cathode Life", Proc. 111th TMS Annual Meeting, Dallas, Texas, Feb. 14-18, 1982, pp. 299-309.
[189]. SORLIE, M., HVISTENDAHL, J. and OYE, H. A., "Early Failure Mechanisms In Aluminium Cell Cathodes", Proc. 122nd TMS Annual Meeting, Denver, Colorado, Feb. 21-25, 1993, pp. 299-308.
[190]. HOMSI, P., "Selection Criteria for Cathode Blocks", Proc. 6th Australasian Aluminium Smelting Workshop, Jul. 13, 1998, pp. 429-450.
[191]. FAANESS, B. M., GRAN, H., SORLIE, M. and OYE, H. A., "Ramming Paste Related Failures in Cathode Linings", Proc. 118th TMS Annual Meeting, Las Vegas, Nevada, Feb. 27 - Mar. 03, 1989, pp. 633-639.
[192]. MENDENHALL, W., WACKERLY, D. D. and SCHEAFFER, R. L., "Mathematical Statistics with Applications", Fourth Edition, PWS-KENT Publishing Company, 1990.
[193]. COLE, D., TERRELL, M. and WOOD, S., "Effect of Iron Source on Vanadium Recovery in Pot Metal", Proc. 124th TMS Annual Meeting, Las Vegas, Feb. 12-16, 1995, pp. 213-215.
[194]. ZHUXIAN, Q., JINGJIANG, L., XIAOLI, C., GRJOTHEIM, K., KVANDE, H. and OYE, H., "Bath Temperature Measurements in Aluminium Electrolysis Cells", Proc. 124th TMS Annual Meeting, Las Vegas, Feb. 12-16, 1995, p. 233
[195]. MADSEN, D. J., "Temperature Measurement and Control in Reduction Cells", Proc. 121st TMS Annual Meeting, San Diego, California, Mar. 01-05, 1992, pp. 453-456.
[196]. VERSTREKEN, P. and BENNINGHOFF, S., "Bath and Liquidus Temperature Sensor for Molten Salts", Proc. 125th TMS Annual Meeting, Anaheim, California, Feb. 04-08, 1996, pp. 437-444.
[197]. REVELL, P., FROST, F. and MATTHEWS, R., "Thermocouple Capability Study for Bath Temperature Measurements", Comalco Aluminium (Bell Bay) Limited, Jun. 09, 1999.
[198]. NAGEL, S. S. and NEEF, M., "Operations Research Methods: Quantitative Applications in the Social Sciences", SAGE Publications, Inc., 1976.
[199]. WINSTON, W. L., "Operations Research: Applications and Algorithms", Second Edition, PWS-KENT Publishing Company, 1987.
[200]. STEWART, W., "Operations Research: Overview", http://www.or.ncsu.edu/overview.dir/overview.html, Nov. 04, 1999.
[201]. DANTZIG, G., "What is Operations Research?", http://www.shocking.com/~batman/illum/swtqw/SilentLinks/OpResearch/OR.html, Nov. 04, 1999.
[202]. FATSEAS, V. A. and VAGG, T. R., "Quantitative Techniques for Managerial Decision Making", Prentice-Hall of Australia Pty Ltd, 1982.
[203]. KARRI, V., "A Course in Advanced Manufacturing: Part - I", Department of Civil and Mechanical Engineering, University of Tasmania, 1994.
[204]. TAHA, H. A., "Operations Research: An Introduction", Fifth Edition, Macmillan Publishing Company, 1992.
[205]. SIMMONS, D. M., "Linear Programming for Operations Research", Holden-Day, Inc., 1972.
[206]. LUENBERGER, D. G., "Introduction to Linear and Nonlinear Programming", Addison-Wesley Publishing Company, Inc., 1973.
[207]. PHILLIPS, D. T., RAVINDRAN, A. and SOLBERG, J. J., "Operations Research: Principles and Practice", John Wiley and Sons, Inc., 1976.
[208]. FRENCH, S., HARTLEY, R., THOMAS, L. C. and WHITE, D. J., "Operational Research Techniques", Edward Arnold Publishers Limited, 1986.
[209]. COHEN, S. S., "Operations Research", S. S. Cohen, 1985.
[210]. CROUCHER, J. S., "Operations Research: A First Course", Pergamon Press (Australia) Pty Ltd, 1980.
[211]. TERSINE, R. J, ALTIMUS, C. A. and DAVIDSON, F, "Problems and Models in Operations Management", Second Edition, Grid Publishing, Inc., 1980.

[212]. VERMA, H. L. and GROSS, C. W., "Introduction to Quantitative Methods: A Managerial Emphasis", John Wiley and Sons, Inc., 1978.
[213]. FRAZER, J. R., "Applied Linear Programming", Prentice-Hall, Inc., 1986.
[214]. JGNIZIO, J. P. and GUPTA, J. N., "Operations Research in Decision Making", Crane, Russak and Company, Inc., 1975.
[215]. HILLIER, F. S. and LIEBERMAN, G. J., "Introduction to Operations Research", McGraw-Hill, Inc., 1987.
[216]. DANTZIG, G. B., "Programming in a Linear Structure", Comptroller, United States Air Force, Washington, D.C., 1948.
[217]. LEVIN, R. I., KIRKPATRICK, C. A. and RUBIN, D. S., "Quantitative Approaches to Management", Fifth Edition, McGraw-Hill, Inc., 1982.
[218]. CROWDER, H., JOHNSON, E. L. and PADBERG, M., "Solving Large-Scale Zero-One Linear Programming Problems", Journ. Operations Research, vol. 31, 1983, pp. 803-834.
[219]. JOHNSON, E. L., KOSTREVA, M. M. and SUHL, U. H., "Solving 0-1 Integer Programming Problems Arising from Large-Scale Planning Models", Journ. Operations Research, vol. 33, 1985, pp. 803-819.
[220]. MITTEN, L. G., "Branch-and-Bound Methods: General Formulation and Properties", Journ. Operations Research, vol. 18, 1970, pp. 24-34.
[221]. SCHRIVER, A., "Theory of Linear and Integer Programming", Wiley, New York, 1986.
[222]. SCHNEIDER, D. I., "An Introduction to Programming Using Visual Basic™", Prentice-Hall, 1995
[223]. MICROSOFT PRESS, "Microsoft® Excel/Visual Basic™ Programmer's Guide", Microsoft Corporation, 1995
[224]. MICROSOFT PRESS, "Visual Basic™ User's Guide: Microsoft® Excel", Microsoft Corporation, 1994
[225]. MICROSOFT PRESS, "Microsoft® Excel 5: Visual Basic™ for Applications Reference", Microsoft Corporation, 1994
[226]. MICROSOFT PRESS, "Microsoft® Excel User's Guide", version 5.0, Microsoft Corporation, 1994
[227]. MICROSOFT PRESS, "Microsoft® Excel 5: Worksheet Function Reference", Microsoft Corporation, 1994
[228]. STEPHENS, R., "Advanced Visual Basic™ Techniques", John Wiley & Sons, Inc., 1997.
[229]. THE AUSTRALIAN, "Business - Metal Prices", Feb. 29, 2000, p.28.
[230]. AURORA ENERGY, "Aurora Customer Electricity Pricing", Tasmania, Australia, Mar. 19, 2000.
[231]. SAS Institute, Inc., "SAS® Language and Procedures: Introduction", Version Six, First Edition, SAS Institute, Inc., 1991.
[232]. SAS Institute, Inc., "SAS® Software", SAS Institute, Inc., Version Six, First Edition, 1991.
[233]. KOHLER, H., "Statistics for Business and Economics", Second Edition, Scott, Foresman and Company, 1988.
[234]. LEWIS, D. E., O'BRIEN, D. T and THAMPAPILLAI, D., "Statistics for Business and Economics", Harcourt Brace Jovanovich Group (Australia) Pty Ltd, 1990.
[235]. LEVIN, R. I., "Statistics for Management", Fourth Edition, Prentice-Hall, Inc., 1987.
[236]. GOLDMAN, R. and WEINBERG, J. S., "Statistics: An Introduction", Prentice-Hall Inc., 1985.
[237]. KOSKO, B., "Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence", Prentice-Hall, Inc., 1992.

APPENDIX A

Neural Network User's Guide

A.1 NEURAL NETWORK USER'S GUIDE

While the particularities of the neural networks included in the accompanying software have been given in Chapter Two, the following documentation is necessary to demonstrate the use of the accompanying neural network programs. All accompanying neural network programs are written in Pascal code and are executable using Turbo Pascal, version 6.0, or later. The neural network programs available in the accompanying software incorporate the neural networks listed in Table A.1.1 and are available in the directory *NN_Prog* in the sub-directory *Programs*. The Pascal program name associated with each neural network is documented in the right-hand side of the table for user reference.

Table A.1.1. Accompanying Neural Networks and Corresponding Pascal Program Name

Neural Network	Program Name
Widrow-Hoff	WH.pas
Backpropagation - 1 hidden layer	BP1.pas
Backpropagation - 2 hidden layers	BP2.pas
Radial Basis Function	RBF.pas
Kohonen	KOH.pas
Radial Basis Function - incorporating Kohonen	RBFKOH.pas
General Regression	GRNN.pas

A.1.1 Opening a Neural Network Program

From the Turbo Pascal menu bar select *File*, then *Open*, as shown in Figure A.1.1(a). Map to the appropriate directory containing the neural network programs, highlight the desired program and click *Open* to open that program, as shown in Figure A.1.1 (b). As a result of this command the selected program is displayed in an editor screen on the user interface.

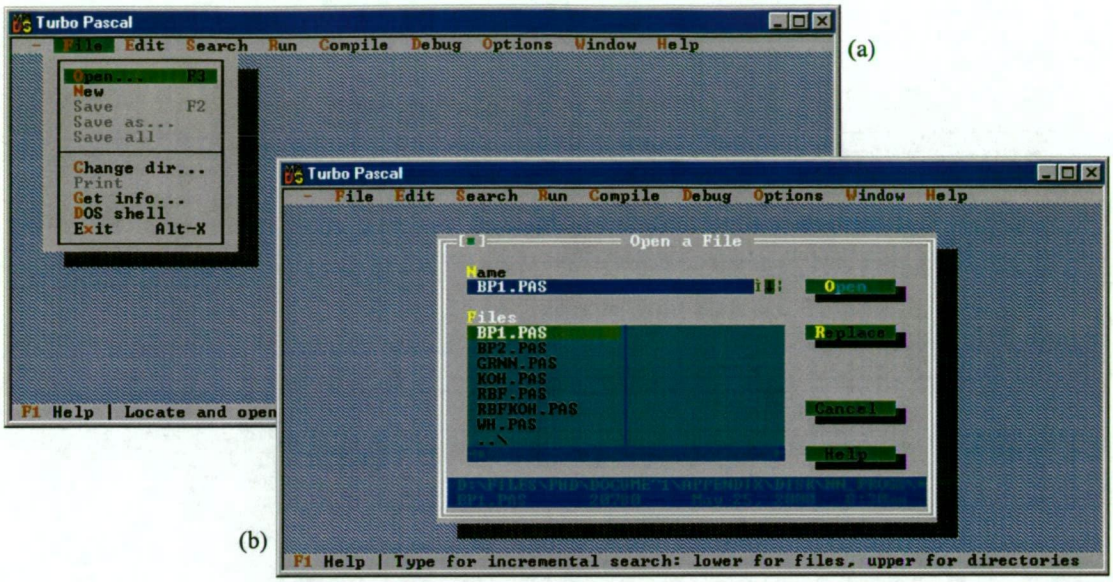


Fig. A.1.1.(a) Activating the Open Dialog Box from Turbo Pascal Menu Bar, and (b) Opening a Neural Network Program in Turbo Pascal

A.1.2 Editing a Neural Network Program

The editor screen, as shown in Figure A.1.2, is displayed on the user interface when a neural network program is opened, enabling the user to specify any necessary program constraints; such as training and test data size, number of iterations required and appropriate directory path where files required for neural network training and testing are stored.

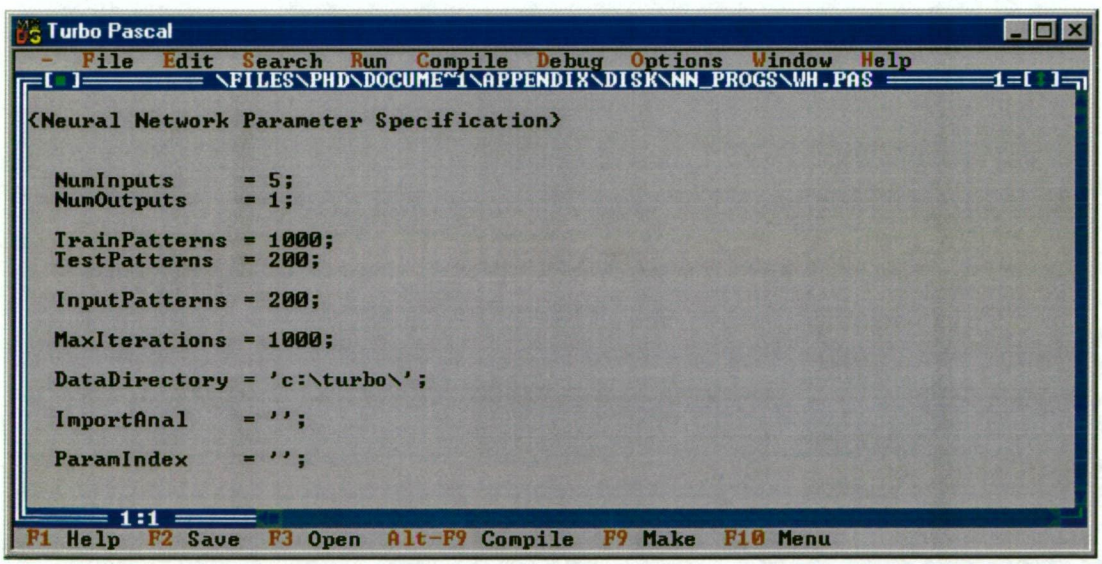


Fig. A.1.2. Turbo Pascal Neural Network Program Editor Screen Displayed on User Interface

The particular constants requiring specification in each neural network program are listed in Table A.1.2, giving a brief description and typical range of values of each constant and the neural network programs to which the constant specification is applicable. It is useful to note that a thorough description of each constant is provided in Chapter Two, which should be referred to for further information.

Table A.1.2. Brief Description and Typical Range of Neural Network Program Constants

Constant Name	Brief Description	Range	Applicable Program(s)
NumInputs	no. of input parameters	1 to 40	all
NumOutputs	no. of output parameters	1 to 10	all
NumHiddenNodes	no. of first hidden layer nodes	2 to 20	BP1, BP2, RBF, RBFKOH.pas
NumHidden2Nodes	no. of second hidden layer nodes	2 to 20	BP2.pas
NumPatternNodes	no. of pattern nodes	5 to 1000	GRNN.pas
NumClusters	no. of classification categories	1 to 10	KOH.pas
TrainPatterns	no. of train data patterns	100 to 2000	all
TestPatterns	no. of test data patterns	10 to 400	WH, BP1, BP2, RBF, RBFKOH, GRNN.pas
InputPatterns	no. of input data patterns	10 to 1000	all
MaxIterations	maximum no. of iterations	100 to 1000	WH, BP1, BP2, RBF, KOH, RBFKOH.pas
Radius	neighbourhood for weight update	1 to 10	KOH.pas
DataDirectory	directory where train and test text files are stored	a:\ to d:\	all
linear	linear activation function	0 or 1	WH, BP1, BP2.pas
sigmoidal	sigmoidal activation function	0 or 1	WH, BP1, BP2.pas
sigma	receptive field width	0.1 to 0.9	RBF, RBFKOH, GRNN.pas

A.1.3 Compiling and Running a Neural Network Program

Following specification of all necessary program constraints, in the first instance select *Compile* from the menu bar to check for program errors as a result of user data entry, as shown in Figure A.1.3(a). While the neural network programs have been rigorously compiled and tested previously, if any compile errors occur as a result of data entry it is typically due to erased semi-colons (;) or punctuation marks (‘ ’) during data entry by the user. Hence, any compiling problems are generally easily

resolved. Following successful compilation of the program, select *Run* from the menu bar, as shown in Figure A.1.3(b), to use the neural network program.

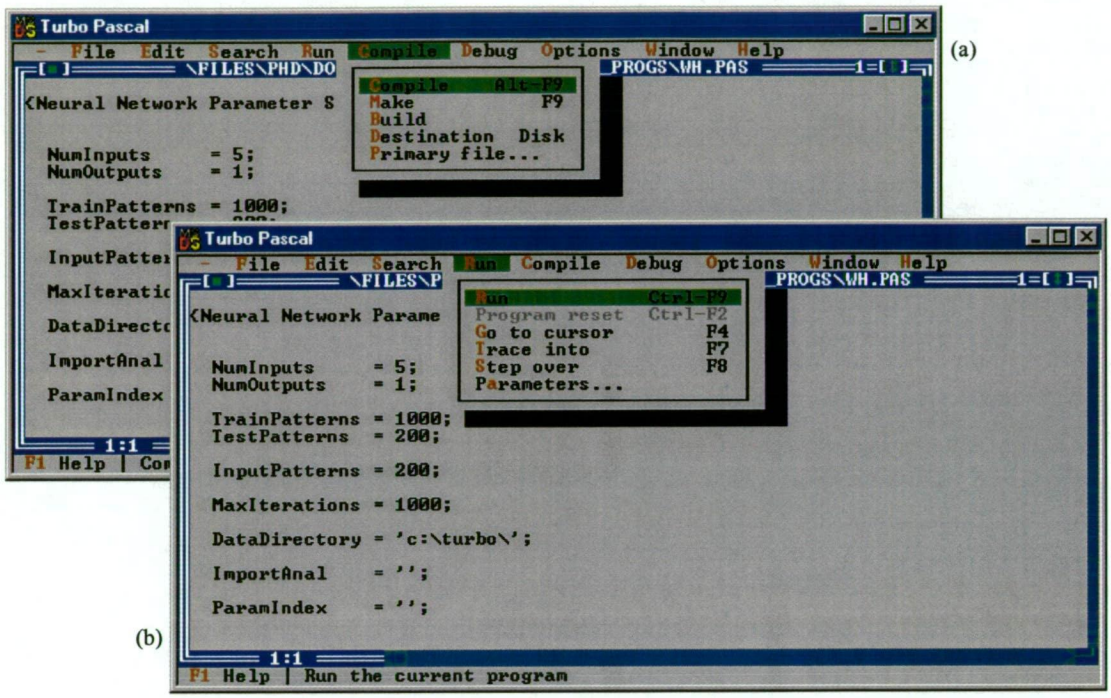


Fig. A.1.3. From Turbo Pascal Menu Bar (a) Compile a Neural Network Program, and (b) Run a Neural Network Program

An output screen is displayed as a result of the *Run* command, prompting the user to select the option of (1) Train and Test the Network, or (2) Use the Network, as shown in Figure A.1.4.

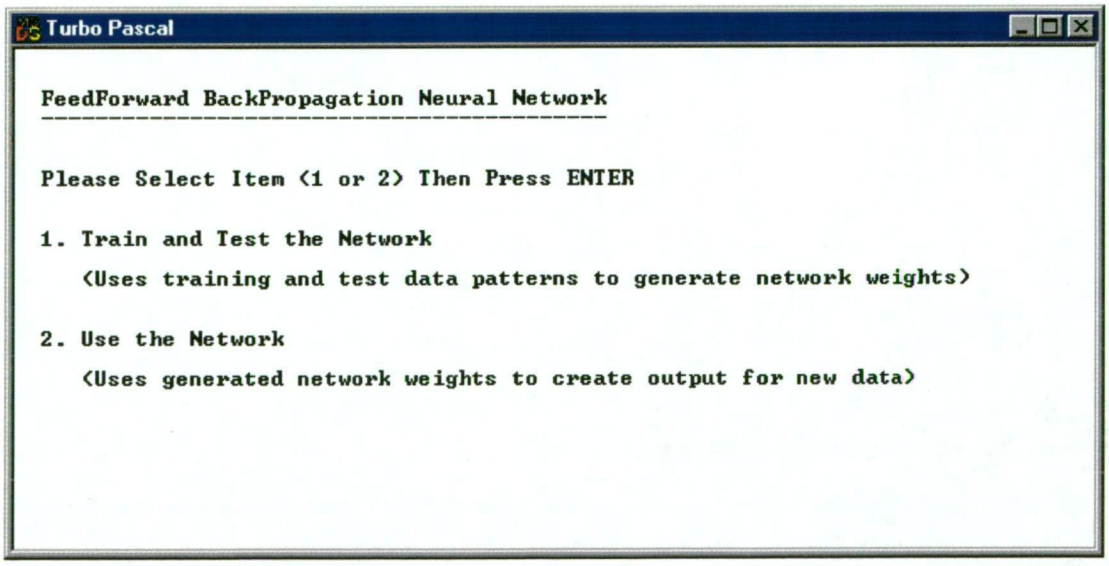


Fig. A.1.4. Output Screen Displayed on User Interface Showing Neural Network Program Options

1. Train and Test the Network - This option is used to train and test a specific neural network. As a result of this command an appropriate weights file is developed for the specified data sets. Further, an error file highlighting train and test RMS error for each iteration is produced. In addition, for each train and test data pattern the actual and predicted values of the output parameter(s) are written to a file and saved to a user specified directory. The computation time required to complete the specified number of iterations is also calculated, written to a file and saved.

There are typically four types of neural network training and testing that would be completed using this option in association with the accompanying neural network programs. They are:

i). Train and test the neural network with all potential input parameters included in the data sets. Vary the neural network architecture and algorithm to establish the minimum RMS error that can be achieved using the specified train and test data sets.

ii). Complete a predictive importance analysis. Using the neural network architecture and algorithm that yields minimum RMS error, as established in (1), for each input parameter sequentially omitted from the train and test data sets, determine the resulting RMS error.

iii). Complete a casual importance analysis. Using the neural network architecture and algorithm that yields minimum RMS error, as established in (1), for each input parameter sequentially varied in the train and test data sets. Determine the resulting RMS error in each instance.

iv). Train and test the neural network with any non-contributing input parameters removed from the data sets. Using the results of the predictive or casual importance analysis, remove any non-contributing input parameters from the train and test data sets. Determine the resulting RMS error in each instance.

It is necessary to note that there are specific data files that are required for neural network training and testing to proceed. In particular, these files contain the train and test data sets. Further, there are specific data files produced as a result of neural network training and testing. These files contain important information summarising neural network training and testing. In particular, the weights matrix, train and test error, actual and predicted values of the output parameter(s) for the train and test data

sets and the computation time required for network training are specified in these files. For each of the listed four types of neural network training and testing that would typically be completed, the required input file names and names of the output files produced are listed in Table A.1.3.

Table A.1.3. Text Files Required for Neural Network Training and Testing and Subsequent Output Files Produced

File Description	(i)	(ii)	(iii)	(iv)
Input files required				
training data patterns (tab-delimited)	trn.txt	pitrn*n*.txt	citrn*n*.txt	trnnon.txt
test data patterns (tab-delimited)	tst.txt	pitst*n*.txt	citst*n*.txt	tstnon.txt
Output files produced				
neural network weights matrix	wts.out	piwts*n*.out	ciwts*n*.out	wtsnon.out
train and test root-mean-square (RMS) error	err.out	pierr*n*.out	cierr*n*.out	errnon.out
actual and predicted output parameter(s) values - train	trn.out	pitrn*n*.out	citrn*n*.out	trnnon.out
actual and predicted output parameter(s) values - test	tst.out	pitst*n*.out	citst*n*.out	tstnon.out
neural network computation time	tim.out	pitim*n*.out	citim*n*.out	timnon.out

There is a single training text file and single test text file required for training and testing the neural network with all potential input parameters included in the data sets. Likewise, for train and testing the neural network with any non-contributing input parameters removed from the data sets, there is a single training text file and single test text file required. However, there are multiple input text files required for training and testing a neural network for an importance analysis. In particular, for i inputs for a predictive importance analysis there are i train data text files and i test data text files required. The train and test data text files required have the identifiers 'pitrn1.txt' and 'pitst1.txt', 'pitrn2.txt' and 'pitst2.txt', ..., 'pitrn*n*.txt' and 'pitst*n*.txt', where n = 1, ..., i. The text files 'pitrn1.txt' and 'pitst1.txt' are unique as they have the first input parameter omitted from the data sets, while all other input and output parameters are maintained, whereas 'pitrn*n*.txt' and 'pitst*n*.txt' are unique as they have the n^{th} input parameter omitted from the data sets, while all other input and output parameters are maintained. Similarly, for a casual importance analysis, there

are i train data text files and i test data text files required. The train and test data text files required have the identifiers 'citrn1.txt' and 'citst1.txt', 'citrn2.txt' and 'citst2.txt', ..., 'citrn$n$.txt' and 'citst$n$.txt', where n = 1, ..., i. The text files 'citrn1.txt' and 'citst1.txt' are unique as they have the first input parameter varied over the bounded range 0.0 to 1.0, while all other input and output parameters are maintained, whereas 'citrn$n$.txt' and 'citst$n$.txt' are unique as they have the n^{th} input parameter varied over the bounded range 0.0 to 1.0, while all other input and output parameters are maintained. All train and test data sets are required to be text files (tab-delimited), as shown in Figure A.1.5.

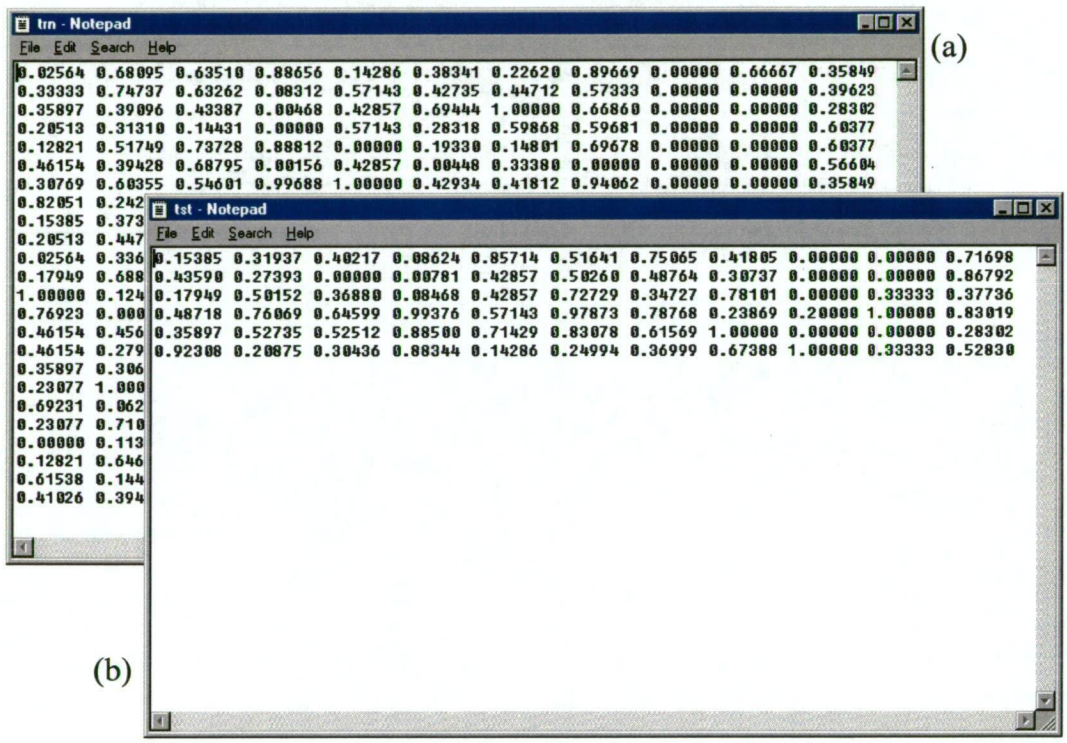


Fig. A.1.5. Data Text Files (10 Input Parameters, 1 Output Parameter, 24 Train Data Patterns and 6 Test Data Patterns) (a) Train, and (b) Test

During neural network training and testing there is specific information displayed on the user interface, as shown in Figure A.1.6. The abbreviation 'Iter' specifies the current iteration being completed, while 'e1' and 'e2' represent the train and test error, respectively, for that iteration. Further, the value displayed in the far right-hand side column represents the learning rate, α. However, this latter feature is not applicable to the KOH.pas or GRNN.pas programs.

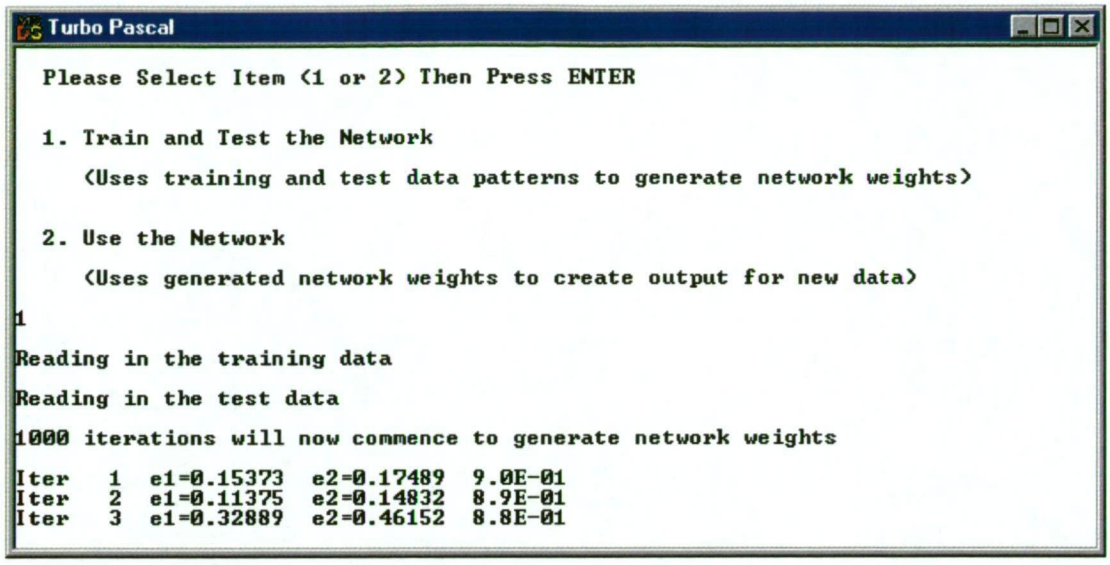


Fig. A.1.6. Output Screen Displayed on User Interface as a Result of Selecting Option (1) from the Neural Network Menu Items

It is necessary to note that computation time associated with each of the neural network programs differs in each instance and can be substantial for some programs. Moreover, computation time is a function of the number of train and test data patterns and iterations specified and model complexity, which is a function of the number of inputs, outputs and processing nodes used. In particular, computation time increases with increasing number of train and test data patterns, iterations and model complexity.

2. Use the Network - This option is used to predict values for each output parameter associated with an application. However, a prerequisite for this command is that neural network training and testing have been previously completed. That is, for the application considered, a neural network model must have previously been trained and tested, with an appropriate weights matrix produced and accessible. The input file required for this command contains values for the corresponding number of input parameters for which the neural network has been previously trained and tested and dummy values (typically 0 or 1 is used) for each output parameter associated with the application. The required identifier for the input file is 'input.txt', which should be a text file (tab-delimited) of the form shown in Figure A.1.5(a) and (b). In addition, the appropriate weight file required should have the identifier 'wts.out'. The output file produced as a consequence of this command is 'output.out', which contains the dummy and predicted values of the output parameter. Figure A.1.7

shows an example output screen displayed on the user interface as a result of this action.

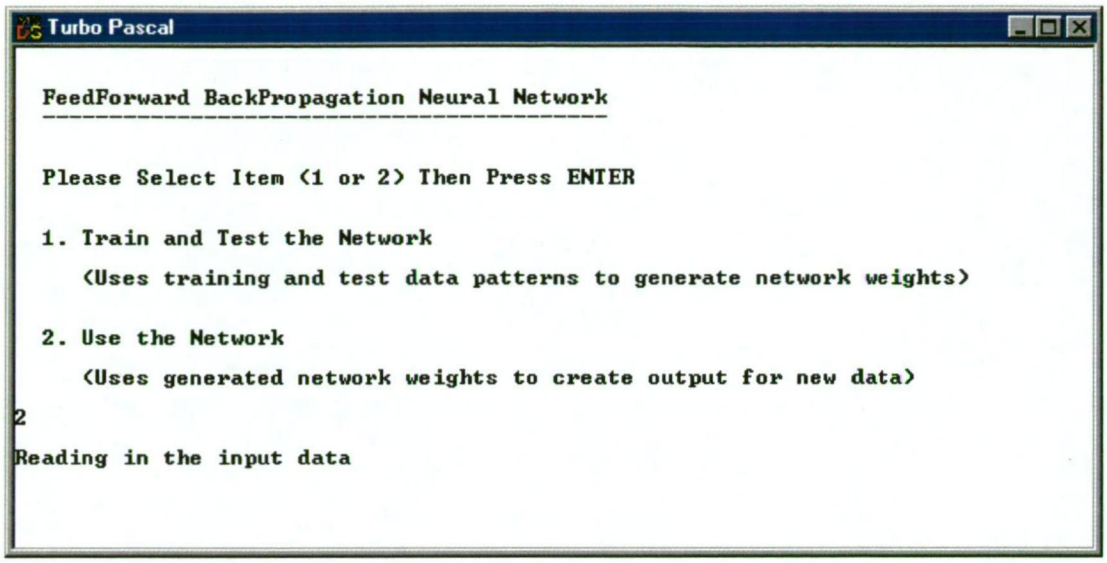


Fig. A.1.7. Output Screen Displayed on User Interface as a Result of Selecting Option (2) from the Neural Network Menu Items

While it is noted that a substantial computation time may be required for training and testing the neural network in some instances, use of the neural network to predict values for a specified input file is significantly faster. In fact, in the majority of instances the 'output.out' is produced and saved to the specified directory in only a few seconds.

A.1.4 Closing a Neural Network Program and Exiting Turbo Pascal

To close the active neural network program, click on the small green button in the top left-hand corner of the editor screen, as shown in Figure A.1.8(a). By default, if program modifications have not been saved prior to closing, the user is prompted, using a message box displayed on the user interface, to save the modifications before the program closes. Exiting Turbo Pascal is completed by selecting *Exit* from the Turbo Pascal menu bar, as shown in Figure A.1.8(b).

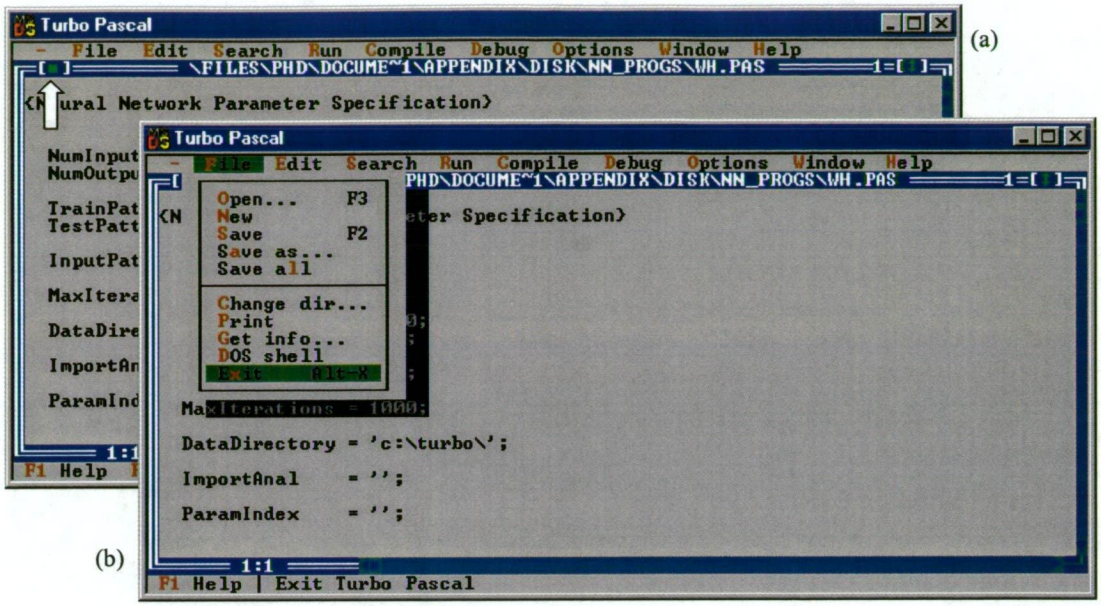


Fig. A.1.8.(a) Closing a Neural Network Program, and (b) Exiting Turbo Pascal Using Menu Bar

A.2 NEURAL NETWORK PROGRAM SOURCE CODE

Formatted descriptive source code is available on the accompanying software for each of the neural network programs listed in Table A.1.1. The source code is available in the directory *NN_Prog* in the sub-directory *Sce_Code*. The source code file has the identifier *NN_Code.doc*.

APPENDIX B

Neural Network Modelling Results for Sensitivity Analysis

TABLE B.1. WH Network RMS Error and Computation Time Behaviour with Changing Network Architecture

Output Layer Activation Function	RMS Error		Computation Time (s)
	Train	Test	
sigmoidal	0.0843	0.0909	129.24
linear	0.0968	0.1032	74.54

TABLE B.2. BP1 Network RMS Error and Computation Time Behaviour with Changing Network Architecture

Hidden Layer Nodes	Hidden Layer Activation Function	Output Layer Activation Function	RMS Error		Computation Time (s)
			Train	Test	
2	sigmoidal	sigmoidal	0.0700	0.0784	315.33
3	sigmoidal	sigmoidal	0.0630	0.0770	428.63
4	sigmoidal	sigmoidal	0.0640	0.0771	574.08
5	sigmoidal	sigmoidal	0.0535	0.0686	684.53
6	sigmoidal	sigmoidal	0.0536	0.0678	799.28
7	sigmoidal	sigmoidal	0.0447	0.0636	943.29
8	sigmoidal	sigmoidal	0.0471	0.0656	1045.23
9	sigmoidal	sigmoidal	0.0564	0.0709	1180.18
10	sigmoidal	sigmoidal	0.0611	0.0780	1286.38
2	sigmoidal	linear	0.0702	0.0803	250.41
3	sigmoidal	linear	0.0700	0.0800	375.85
4	sigmoidal	linear	0.0630	0.0792	501.05
5	sigmoidal	linear	0.0656	0.0788	595.63
6	sigmoidal	linear	0.0675	0.0831	705.27
7	sigmoidal	linear	0.0553	0.0742	857.90
8	sigmoidal	linear	0.0598	0.0779	985.67
9	sigmoidal	linear	0.0616	0.0791	1054.43
10	sigmoidal	linear	0.0623	0.0791	1230.45

TABLE B.3.1. BP2 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers

Hidden Layer 1 Nodes	Hidden Layer 2 Nodes	RMS Error		Computation Time (s)
		Train	Test	
2	2	0.0859	0.0934	563.54
2	3	0.0778	0.0871	733.37
2	4	0.1190	0.1280	949.63
2	5	0.0885	0.0964	1120.97
2	6	0.0846	0.0980	1288.78
2	7	0.0959	0.1060	1476.34
2	8	0.1430	0.1536	1669.03
2	9	0.1435	0.1543	1772.80
2	10	0.1443	0.1552	1876.80

3	2	0.0688	0.0852	748.29
3	3	0.0829	0.0885	918.56
3	4	0.1417	0.1524	1120.64
3	5	0.0722	0.0813	1298.61
3	6	0.1095	0.1214	1469.56
3	7	0.1423	0.1531	1658.23
3	8	0.0807	0.0901	1842.67
3	9	0.0957	0.1044	1957.31
3	10	0.0888	0.0975	2061.58
4	2	0.0902	0.1038	924.62
4	3	0.0943	0.1088	1085.47
4	4	0.0781	0.0887	1284.67
4	5	0.0803	0.0921	1483.56
4	6	0.0690	0.0804	1649.81
4	7	0.0634	0.0767	1864.27
4	8	0.0658	0.0820	2041.07
4	9	0.0712	0.0805	2141.31
4	10	0.0658	0.0823	2258.94
5	2	0.0513	0.0702	1108.65
5	3	0.0561	0.0746	1284.56
5	4	0.0520	0.0720	1479.62
5	5	0.0525	0.0703	1668.54
5	6	0.0520	0.0670	1833.57
5	7	0.0513	0.0694	2053.61
5	8	0.0600	0.0777	2225.64
5	9	0.0596	0.0758	2331.89
5	10	0.0664	0.0833	2431.50
6	2	0.0569	0.0732	1295.64
6	3	0.0388	0.0603	1459.62
6	4	0.0500	0.0707	1668.94
6	5	0.0508	0.0683	1852.73
6	6	0.0481	0.0699	2031.86
6	7	0.0562	0.0771	2219.84
6	8	0.0531	0.0741	2408.92
6	9	0.0621	0.0740	2550.61
6	10	0.0460	0.0672	2635.70
7	2	0.0630	0.0748	1476.94
7	3	0.0565	0.0729	1642.58
7	4	0.0481	0.0696	1861.72
7	5	0.0632	0.0791	2039.81
7	6	0.0434	0.0646	2215.53
7	7	0.0462	0.0694	2421.08
7	8	0.0510	0.0696	2592.81
7	9	0.0583	0.0724	2708.51
7	10	0.0527	0.0668	2833.76
8	2	0.0448	0.0656	1660.48
8	3	0.0519	0.0703	1829.47
8	4	0.0442	0.0655	2046.59
8	5	0.0399	0.0628	2232.15
8	6	0.0460	0.0700	2394.64
8	7	0.0460	0.0678	2604.82
8	8	0.0430	0.0663	2775.79
8	9	0.0483	0.0686	2891.47
8	10	0.0432	0.0651	2984.50
9	2	0.0508	0.0719	1846.58
9	3	0.0489	0.0696	2014.64
9	4	0.0475	0.0686	2228.53
9	5	0.0462	0.0682	2409.61
9	6	0.0494	0.0731	2584.73

9	7	0.0485	0.0697	2794.16
9	8	0.0495	0.0727	2964.81
9	9	0.0487	0.0699	3078.53
9	10	0.0505	0.0726	3174.28
10	2	0.0512	0.0724	2031.49
10	3	0.0511	0.0718	2187.65
10	4	0.0480	0.0696	2412.87
10	5	0.0485	0.0704	2641.80
10	6	0.0462	0.0687	2767.79
10	7	0.0457	0.0685	2987.87
10	8	0.0495	0.0731	3148.57
10	9	0.0485	0.0713	3257.41
10	10	0.0467	0.0689	3342.86

TABLE B.3.2. BP2 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal and Linear Activation Function in the Hidden and Output Layers, Respectively

2	2	0.0812	0.0998	458.23
2	3	0.0791	0.0965	584.59
2	4	0.0785	0.0964	761.28
2	5	0.1428	0.1584	895.34
2	6	0.0954	0.1112	1038.94
2	7	0.0998	0.1137	1185.37
2	8	0.1324	0.1486	1337.82
2	9	0.1284	0.1349	1420.61
2	10	0.0875	0.1000	1504.89
3	2	0.0874	0.1015	651.28
3	3	0.0798	0.0954	772.56
3	4	0.0842	0.0998	945.21
3	5	0.0956	0.1046	1081.54
3	6	0.0782	0.0915	1229.50
3	7	0.0684	0.0805	1371.81
3	8	0.0697	0.0818	1529.64
3	9	0.0789	0.0890	1611.99
3	10	0.0804	0.0943	1690.19
4	2	0.0687	0.0841	832.91
4	3	0.0841	0.0993	961.57
4	4	0.0795	0.0916	1134.20
4	5	0.0815	0.0942	1264.80
4	6	0.0796	0.0938	1418.61
4	7	0.0684	0.0794	1542.68
4	8	0.0691	0.0807	1719.61
4	9	0.0677	0.0814	1804.70
4	10	0.0705	0.0853	1875.49
5	2	0.0654	0.0817	1025.41
5	3	0.0605	0.0762	1149.08
5	4	0.0598	0.0739	1327.29
5	5	0.0571	0.0716	1467.82
5	6	0.0584	0.0741	1582.04
5	7	0.0561	0.0724	1748.55
5	8	0.0542	0.0705	1898.73
5	9	0.0609	0.0763	1986.42
5	10	0.0618	0.0780	2060.18
6	2	0.0571	0.0741	1207.59
6	3	0.0516	0.0692	1333.95
6	4	0.0498	0.0665	1510.64
6	5	0.0452	0.0633	1644.70
6	6	0.0472	0.0648	1788.38
6	7	0.0469	0.0637	1934.73

6	8	0.0482	0.0667	2087.18
6	9	0.0461	0.0618	2169.97
6	10	0.0504	0.0660	2245.48
7	2	0.0512	0.0700	1387.52
7	3	0.0619	0.0794	1518.34
7	4	0.0624	0.0796	1694.52
7	5	0.0598	0.0747	1831.45
7	6	0.0495	0.0714	1973.45
7	7	0.0564	0.0742	2128.64
7	8	0.0642	0.0818	2276.51
7	9	0.0617	0.0791	2377.10
7	10	0.0537	0.0729	2435.81
8	2	0.0512	0.0724	1572.68
8	3	0.0541	0.0761	1709.86
8	4	0.0499	0.0718	1882.69
8	5	0.0537	0.0746	2019.31
8	6	0.0581	0.0787	2158.73
8	7	0.0562	0.0771	2321.94
8	8	0.0537	0.0749	2465.28
8	9	0.0618	0.0800	2568.48
8	10	0.0593	0.0824	2628.71
9	2	0.0584	0.0841	1758.62
9	3	0.0641	0.0824	1885.27
9	4	0.0637	0.0818	2067.61
9	5	0.0594	0.0843	2218.74
9	6	0.0582	0.0810	2342.08
9	7	0.0637	0.0838	2509.67
9	8	0.0605	0.0824	2671.09
9	9	0.0572	0.0811	2749.06
9	10	0.0596	0.0834	2817.09
10	2	0.0615	0.0812	1945.82
10	3	0.0589	0.0818	2079.64
10	4	0.0637	0.0831	2264.81
10	5	0.0581	0.0812	2408.91
10	6	0.0637	0.0828	2547.97
10	7	0.0675	0.0854	2696.85
10	8	0.0594	0.0811	2857.57
10	9	0.0566	0.0798	2935.09
10	10	0.0687	0.0872	3047.61

TABLE B.4. RBF Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1414	0.1518	115.40
0.1	10	0.1406	0.1515	225.64
0.1	20	0.1378	0.1497	441.51
0.1	40	0.1351	0.1481	883.25
0.1	50	0.1324	0.1476	1086.42
0.1	60	0.1351	0.1497	1290.35
0.1	80	0.1330	0.1489	1659.83
0.2	5	0.1406	0.1492	114.99
0.2	10	0.1390	0.1482	224.70
0.2	20	0.1347	0.1448	441.62
0.2	40	0.1295	0.1411	884.34
0.2	50	0.1289	0.1409	1085.91
0.2	60	0.1276	0.1406	1290.64

0.2	80	0.1248	0.1394	1658.92
0.3	5	0.1293	0.1298	115.32
0.3	10	0.1239	0.1290	225.42
0.3	20	0.1136	0.1234	441.14
0.3	40	0.1030	0.1120	884.91
0.3	50	0.1108	0.1093	1084.76
0.3	60	0.0974	0.1061	1289.62
0.3	80	0.0925	0.1048	1660.27
0.4	5	0.1175	0.1204	115.71
0.4	10	0.1096	0.1185	224.91
0.4	20	0.0980	0.1113	440.96
0.4	40	0.0878	0.0970	884.59
0.4	50	0.0826	0.0957	1083.81
0.4	60	0.0806	0.0951	1289.95
0.4	80	0.0769	0.0967	1659.36
0.5	5	0.1111	0.1157	114.87
0.5	10	0.0992	0.1100	224.89
0.5	20	0.0889	0.1027	440.98
0.5	40	0.0784	0.0891	883.94
0.5	50	0.0740	0.0891	1085.32
0.5	60	0.0731	0.0890	1290.31
0.5	80	0.0706	0.0912	1660.60
0.6	5	0.1080	0.1137	115.80
0.6	10	0.0920	0.1036	225.06
0.6	20	0.0840	0.0977	441.03
0.6	40	0.0734	0.0852	884.37
0.6	50	0.0683	0.0833	1084.60
0.6	60	0.0739	0.0884	1290.58
0.6	80	0.0743	0.0925	1660.42
0.7	5	0.1065	0.1130	115.26
0.7	10	0.0854	0.0989	225.18
0.7	20	0.0814	0.0953	442.32
0.7	40	0.0739	0.0862	885.14
0.7	50	0.0741	0.0881	1084.73
0.7	60	0.0761	0.0918	1291.52
0.7	80	0.0859	0.1027	1660.58
0.8	5	0.1060	0.1129	116.07
0.8	10	0.0838	0.0957	225.34
0.8	20	0.0850	0.0947	441.56
0.8	40	0.0732	0.0864	884.61
0.8	50	0.0732	0.0865	1085.42
0.8	60	0.0796	0.0923	1291.18
0.8	80	0.0865	0.1008	1661.48
0.9	5	0.1057	0.1130	115.53
0.9	10	0.0817	0.0937	224.96
0.9	20	0.0802	0.0946	441.87
0.9	40	0.0782	0.0871	884.62
0.9	50	0.0814	0.0924	1085.17
0.9	60	0.0840	0.0979	1290.94
0.9	80	0.0912	0.1088	1660.07

TABLE B.5. RBFKOH Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1425	0.1526	369.52
0.1	10	0.1405	0.1523	704.82
0.1	20	0.1378	0.1495	1342.54
0.1	40	0.1328	0.1502	2502.18
0.1	50	0.1314	0.1497	3089.93
0.1	60	0.1263	0.1453	3631.70
0.1	80	0.1281	0.1453	4693.52
0.2	5	0.1400	0.1518	369.62
0.2	10	0.1376	0.1488	703.78
0.2	20	0.1316	0.1429	1343.15
0.2	40	0.1247	0.1449	2502.26
0.2	50	0.1221	0.1418	3089.96
0.2	60	0.1111	0.1305	3630.92
0.2	80	0.1111	0.1305	4693.08
0.3	5	0.1283	0.1418	370.00
0.3	10	0.1241	0.1342	704.55
0.3	20	0.1096	0.1204	1342.96
0.3	40	0.1020	0.1235	2502.53
0.3	50	0.0961	0.1137	3089.93
0.3	60	0.0891	0.1090	3632.17
0.3	80	0.0891	0.1090	4692.70
0.4	5	0.1224	0.1361	370.26
0.4	10	0.1089	0.1196	704.33
0.4	20	0.0954	0.1060	1343.65
0.4	40	0.0887	0.1092	2502.90
0.4	50	0.0842	0.1005	3090.63
0.4	60	0.0789	0.1000	3631.82
0.4	80	0.0789	0.1000	4692.16
0.5	5	0.1160	0.1299	369.41
0.5	10	0.0967	0.1080	704.09
0.5	20	0.0875	0.0977	1343.98
0.5	40	0.0803	0.0996	2502.53
0.5	50	0.0762	0.0917	3089.80
0.5	60	0.0743	0.0926	3631.73
0.5	80	0.0743	0.0926	4692.42
0.6	5	0.1118	0.1261	369.77
0.6	10	0.0887	0.1004	704.76
0.6	20	0.0834	0.0933	1343.85
0.6	40	0.0761	0.0942	2501.80
0.6	50	0.0691	0.0832	3089.58
0.6	60	0.0728	0.0877	3630.98
0.6	80	0.0728	0.0877	4693.45
0.7	5	0.1094	0.1241	370.34
0.7	10	0.0840	0.0959	704.87
0.7	20	0.0810	0.0919	1342.51
0.7	40	0.0760	0.0940	2502.06
0.7	50	0.0665	0.0798	3090.51
0.7	60	0.0711	0.0865	3631.42
0.7	80	0.0711	0.0865	4692.77
0.8	5	0.1080	0.1232	369.30
0.8	10	0.0814	0.0933	704.01
0.8	20	0.0825	0.0928	1343.66
0.8	40	0.0704	0.0844	2502.37
0.8	50	0.0734	0.0857	3089.97

0.8	60	0.0712	0.0858	3631.92
0.8	80	0.0712	0.0858	4692.58
0.9	5	0.1074	0.1228	369.49
0.9	10	0.0800	0.0920	704.40
0.9	20	0.0843	0.0950	1343.09
0.9	40	0.0715	0.0877	2501.63
0.9	50	0.0793	0.0937	3089.62
0.9	60	0.0781	0.0929	3631.55
0.9	80	0.0781	0.0929	4693.19

TABLE B.6. GRNN RMS Error and Computation Time Behaviour with Changing Network Architecture using an Exponential and Linear Activation Function in the Pattern and Summation Layers, Respectively

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	10	0.1349	0.1312	0.22
0.1	50	0.1083	0.1102	0.55
0.1	100	0.0984	0.1147	1.26
0.1	200	0.0858	0.1131	3.24
0.1	400	0.0681	0.1073	9.61
0.1	600	0.0502	0.0961	18.95
0.1	800	0.0387	0.0958	31.47
0.1	1000	0.0364	0.0952	46.43
0.2	10	0.1184	0.1161	0.21
0.2	50	0.0971	0.1027	0.54
0.2	100	0.0942	0.1089	1.24
0.2	200	0.0799	0.1021	3.28
0.2	400	0.0651	0.1009	9.79
0.2	500	0.0565	0.0979	14.52
0.2	600	0.0483	0.0925	19.08
0.2	800	0.0393	0.0951	31.46
0.2	1000	0.0375	0.0959	46.82
0.3	10	0.1094	0.1099	0.22
0.3	50	0.0960	0.1050	0.59
0.3	100	0.0984	0.1130	1.24
0.3	200	0.0852	0.1045	3.74
0.3	400	0.0752	0.1066	9.53
0.3	600	0.0666	0.1035	18.92
0.3	800	0.0647	0.1062	31.56
0.3	1000	0.0650	0.1065	46.31
0.5	10	0.1062	0.1117	0.24
0.5	50	0.1051	0.1169	0.53
0.5	100	0.1102	0.1245	1.28
0.5	200	0.1025	0.1179	3.22
0.5	400	0.1020	0.1198	9.50
0.5	600	0.1020	0.1195	18.12
0.5	800	0.1035	0.1211	31.18
0.5	1000	0.1043	0.1217	46.61
0.7	10	0.1099	0.1179	0.21
0.7	50	0.1137	0.1256	0.51
0.7	100	0.1185	0.1314	1.23
0.7	200	0.1136	0.1262	3.31
0.7	400	0.1140	0.1274	9.65
0.7	600	0.1145	0.1276	18.99
0.7	800	0.1157	0.1288	31.82
0.7	1000	0.1163	0.1289	45.99
0.9	10	0.1142	0.1233	0.23
0.9	50	0.1195	0.1311	0.54

0.9	100	0.1234	0.1356	1.28
0.9	200	0.1195	0.1312	3.66
0.9	400	0.1201	0.1321	9.54
0.9	600	0.1205	0.1324	18.78
0.9	800	0.1215	0.1334	32.07
0.9	1000	0.1220	0.1334	46.92

APPENDIX C

Neural Network Modelling Results for Industrial Applications

TABLE C.1. WH Network RMS Error and Computation Time Behaviour with Changing Network Architecture for Electrolyte Additive Prediction Application

Output Layer Activation Function	RMS Error		Computation Time (s)
	Train	Test	
sigmoidal	0.1073	0.1140	321.81
linear	0.1086	0.1157	245.90

TABLE C.2. BP1 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application

Hidden Layer Nodes	RMS Error		Computation Time (s)
	Train	Test	
2	0.1004	0.1095	656.18
3	0.1006	0.1102	815.75
4	0.0748	0.0834	962.90
5	0.0755	0.0853	1139.62
6	0.0741	0.0814	1315.94
7	0.0789	0.0821	1446.39
8	0.0737	0.0800	1610.03
9	0.0779	0.0860	1759.21
10	0.0782	0.0835	1920.56
15	0.0812	0.0838	2635.18
20	0.0855	0.0864	3361.77

TABLE C.3. BP2 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application

Hidden Layer 1 Nodes	Hidden Layer 2 Nodes	RMS Error		Computation Time (s)
		Train	Test	
2	2	0.0873	0.0881	1068.51
2	3	0.0869	0.0877	1248.90
2	4	0.0860	0.0871	1562.02
2	5	0.0871	0.0917	1740.67
2	6	0.0973	0.1040	1911.96
2	7	0.0962	0.0981	2108.96
2	8	0.0910	0.0905	2407.89
2	9	0.1011	0.1016	2574.04
2	10	0.0991	0.0998	2707.93
3	2	0.0857	0.0874	1251.45
3	3	0.0838	0.0851	1405.84
3	4	0.0748	0.0751	1716.54
3	5	0.0753	0.0762	1889.93
3	6	0.0851	0.0862	2057.46
3	7	0.0871	0.0872	2246.42

3	8	0.0824	0.0861	2534.08
3	9	0.0847	0.0893	2663.45
3	10	0.0950	0.0955	2810.88
4	2	0.0730	0.0773	1438.28
4	3	0.0743	0.0792	1606.41
4	4	0.0784	0.0787	1891.98
4	5	0.0756	0.0792	2084.95
4	6	0.0718	0.0817	2284.77
4	7	0.0872	0.0904	2460.20
4	8	0.0842	0.0955	2730.60
4	9	0.0774	0.0871	2922.86
4	10	0.0741	0.0827	3068.44
5	2	0.0732	0.0765	1636.11
5	3	0.0716	0.0764	1805.82
5	4	0.0688	0.0885	2132.60
5	5	0.0714	0.0813	2324.51
5	6	0.0759	0.0800	2494.58
5	7	0.0751	0.0792	2669.18
5	8	0.0664	0.0759	2962.73
5	9	0.0706	0.0819	3142.18
5	10	0.0883	0.0888	3316.67
6	2	0.0683	0.0745	1848.73
6	3	0.0672	0.0738	2037.04
6	4	0.0668	0.0739	2336.52
6	5	0.0673	0.0807	2498.28
6	6	0.0662	0.0855	2663.48
6	7	0.0691	0.0751	2848.26
6	8	0.0688	0.0737	3152.59
6	9	0.0693	0.0780	3321.84
6	10	0.0715	0.0802	3508.62
7	2	0.0744	0.0747	2005.52
7	3	0.0651	0.0748	2174.40
7	4	0.0638	0.0749	2511.26
7	5	0.0701	0.0757	2701.03
7	6	0.0649	0.0765	2888.35
7	7	0.0653	0.0759	3080.49
7	8	0.0670	0.0820	3375.77
7	9	0.0692	0.0790	3516.58
7	10	0.0728	0.0813	3664.72
8	2	0.0705	0.0735	2231.39
8	3	0.0682	0.0750	2407.95
8	4	0.0654	0.0775	2705.97
8	5	0.0635	0.0762	2921.77
8	6	0.0615	0.0719	3089.06
8	7	0.0649	0.0738	3268.41
8	8	0.0653	0.0724	3545.72
8	9	0.0681	0.0764	3677.64
8	10	0.0694	0.0771	3843.15
9	2	0.0694	0.0737	2390.18
9	3	0.0674	0.0748	2576.72
9	4	0.0668	0.0769	2893.50
9	5	0.0657	0.0750	3087.36
9	6	0.0638	0.0728	3267.16
9	7	0.0671	0.0752	3433.97
9	8	0.0639	0.0732	3711.90
9	9	0.0745	0.0801	3911.85
9	10	0.0712	0.0805	4070.33
10	2	0.0651	0.0779	2590.47
10	3	0.0672	0.0740	2810.83

10	4	0.0666	0.0725	3107.64
10	5	0.0650	0.0751	3278.08
10	6	0.0741	0.0734	3482.67
10	7	0.0684	0.0782	3684.91
10	8	0.0726	0.0804	3984.70
10	9	0.0757	0.0813	4107.59
10	10	0.0762	0.0825	4263.03

TABLE C.4. RBF Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1034	0.1095	292.96
0.1	10	0.1008	0.1038	731.42
0.1	20	0.0998	0.1011	1455.04
0.1	40	0.0989	0.1006	3062.73
0.1	60	0.0974	0.1005	4617.56
0.1	80	0.0972	0.1014	6117.59
0.1	100	0.0971	0.1018	7582.28
0.3	5	0.0932	0.1041	292.04
0.3	10	0.0922	0.0955	728.45
0.3	20	0.0886	0.0958	1457.16
0.3	40	0.0875	0.0952	3063.74
0.3	60	0.0871	0.0961	4616.37
0.3	80	0.0863	0.0965	6118.29
0.3	100	0.0861	0.0972	7582.95
0.5	5	0.0833	0.0909	291.14
0.5	10	0.0804	0.0847	727.97
0.5	20	0.0790	0.0831	1459.70
0.5	40	0.0759	0.0820	3065.05
0.5	60	0.0748	0.0824	4614.42
0.5	80	0.0747	0.0831	6118.56
0.5	100	0.0739	0.0835	7584.31
0.7	5	0.0728	0.0815	290.96
0.7	10	0.0695	0.0768	726.07
0.7	20	0.0658	0.0746	1455.55
0.7	40	0.0649	0.0748	3061.12
0.7	60	0.0651	0.0751	4619.93
0.7	80	0.0644	0.0760	6120.48
0.7	100	0.0640	0.0764	7581.71
0.8	5	0.0701	0.0798	291.37
0.8	10	0.0666	0.0754	725.08
0.8	20	0.0629	0.0746	1451.93
0.8	40	0.0628	0.0738	3058.17
0.8	60	0.0626	0.0741	4621.64
0.8	80	0.0621	0.0746	6123.61
0.8	100	0.0622	0.0752	7582.18
0.9	5	0.0693	0.0793	288.64
0.9	10	0.0647	0.0744	727.49
0.9	20	0.0635	0.0730	1457.08
0.9	30	0.0632	0.0732	2195.14
0.9	40	0.0630	0.0732	3059.46
0.9	50	0.0623	0.0734	3912.38
0.9	60	0.0614	0.0735	4618.55
0.9	80	0.0607	0.0743	6115.72
0.9	100	0.0611	0.0749	7587.91
1.0	5	0.0685	0.0790	293.51

1.0	10	0.0635	0.0740	723.10
1.0	20	0.0613	0.0736	1448.62
1.0	40	0.0624	0.0741	3060.94
1.0	60	0.0627	0.0745	4618.33
1.0	80	0.0629	0.0750	6125.75
1.0	100	0.0632	0.0762	7580.08

TABLE C.5. RBFKOH Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1047	0.1098	1220.91
0.1	10	0.1014	0.1042	1798.61
0.1	20	0.0999	0.1014	2953.33
0.1	40	0.0984	0.1001	5275.89
0.1	60	0.0980	0.1007	7555.79
0.1	80	0.0982	0.1013	9760.75
0.1	100	0.0977	0.1025	11986.49
0.3	5	0.0837	0.0946	1220.88
0.3	10	0.0804	0.0932	1793.49
0.3	20	0.0776	0.0918	2952.36
0.3	40	0.0762	0.0924	5281.01
0.3	60	0.0766	0.0923	7547.44
0.3	80	0.0762	0.0935	9768.51
0.3	100	0.0761	0.0943	11989.52
0.5	5	0.0739	0.0915	1217.82
0.5	10	0.0725	0.0884	1798.60
0.5	20	0.0709	0.0823	2951.91
0.5	40	0.0695	0.0835	5281.93
0.5	60	0.0694	0.0832	7551.18
0.5	80	0.0690	0.0841	9762.46
0.5	100	0.0688	0.0868	11988.57
0.7	5	0.0714	0.0812	1218.72
0.7	10	0.0667	0.0753	1795.03
0.7	20	0.0648	0.0739	2950.65
0.7	40	0.0637	0.0738	5275.60
0.7	60	0.0635	0.0742	7559.92
0.7	80	0.0638	0.0754	9771.83
0.7	100	0.0637	0.0761	11988.08
0.8	5	0.0696	0.0797	1219.37
0.8	10	0.0658	0.0743	1793.45
0.8	20	0.0625	0.0724	2946.44
0.8	40	0.0622	0.0728	5266.72
0.8	60	0.0623	0.0734	7556.98
0.8	80	0.0625	0.0750	9773.98
0.8	100	0.0624	0.0751	11992.82
0.9	5	0.0627	0.0784	1219.24
0.9	10	0.0615	0.0752	1792.67
0.9	20	0.0607	0.0716	2955.95
0.9	30	0.0604	0.0724	4050.76
0.9	40	0.0596	0.0739	5272.76
0.9	50	0.0601	0.0742	6471.69
0.9	60	0.0600	0.0745	7552.91
0.9	80	0.0602	0.0761	9759.27
0.9	100	0.0601	0.0786	12000.11
1.0	5	0.0677	0.0794	1222.54
1.0	10	0.0669	0.0751	1791.47

1.0	20	0.0622	0.0728	2949.65
1.0	40	0.0617	0.0734	5273.16
1.0	60	0.0628	0.0749	7552.39
1.0	80	0.0625	0.0757	9775.04
1.0	100	0.0631	0.0771	11988.80

TABLE C.6. GRNN RMS Error and Computation Time Behaviour with Changing Network Architecture using an Exponential and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Additive Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	10	0.1235	0.1370	1.65
0.1	50	0.0803	0.0889	5.71
0.1	100	0.0734	0.0828	9.93
0.1	200	0.0692	0.0781	17.79
0.1	400	0.0568	0.0717	24.83
0.1	500	0.0547	0.0708	35.21
0.1	600	0.0549	0.0705	41.75
0.1	800	0.0546	0.0711	51.64
0.1	1000	0.0539	0.0718	59.86
0.1	1365	0.0532	0.0723	77.29
0.2	10	0.1208	0.1335	1.59
0.2	50	0.0811	0.0913	5.34
0.2	100	0.0770	0.0867	9.38
0.2	200	0.0749	0.0834	16.85
0.2	400	0.0679	0.0794	23.60
0.2	600	0.0665	0.0781	42.19
0.2	800	0.0777	0.0783	51.84
0.2	1000	0.0769	0.0785	60.07
0.2	1365	0.0751	0.0792	76.10
0.3	10	0.1204	0.1331	1.67
0.3	50	0.0847	0.0951	5.99
0.3	100	0.0813	0.0908	10.14
0.3	200	0.0800	0.0883	18.36
0.3	400	0.0758	0.0851	25.47
0.3	600	0.0746	0.0841	40.82
0.3	800	0.0732	0.0841	52.51
0.3	1000	0.0721	0.0843	58.38
0.3	1365	0.0718	0.0845	76.94
0.5	10	0.1205	0.1333	1.58
0.5	50	0.0894	0.0998	5.94
0.5	100	0.0863	0.0956	10.21
0.5	200	0.0856	0.0942	16.83
0.5	400	0.0830	0.0914	25.35
0.5	600	0.0820	0.0907	42.36
0.5	800	0.0817	0.0909	50.64
0.5	1000	0.0815	0.0911	58.81
0.5	1365	0.0812	0.0915	76.62
0.7	10	0.1207	0.1335	1.72
0.7	50	0.0919	0.1023	5.43
0.7	100	0.0887	0.0982	9.55
0.7	200	0.0885	0.0974	18.07
0.7	400	0.0862	0.0947	23.60
0.7	600	0.0854	0.0942	42.93
0.7	800	0.0849	0.0943	52.71
0.7	1000	0.0845	0.0945	60.39
0.7	1365	0.0840	0.0952	76.42
0.9	10	0.1209	0.1337	1.72

0.9	50	0.0933	0.1038	6.03
0.9	100	0.0902	0.0996	9.81
0.9	200	0.0903	0.0994	18.47
0.9	400	0.0881	0.0968	25.28
0.9	600	0.0874	0.0964	42.50
0.9	800	0.0871	0.0971	52.35
0.9	1000	0.0868	0.0973	60.72
0.9	1365	0.0861	0.0976	78.19

TABLE C.7. WH Network Predictive Importance Results for Sigmoidal Activation Function in the Output Layer for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.1073	0.1140	-	321.81
target bath temperature	0.1089	0.1208	15.665	299.36
bath temperature	0.1157	0.1160	11.843	298.92
bath height	0.1073	0.1140	0.631	299.30
bath resistivity	0.1157	0.1309	18.541	300.83
emf	0.1072	0.1136	1.419	299.15
AlF_3 addition (t-1)	0.1072	0.1150	5.307	299.71
Na_2CO_3 addition (t-1)	0.1073	0.1146	2.947	301.07
cell power	0.1072	0.1132	2.842	300.52
cell age	0.1073	0.1137	0.828	298.44
F content	0.1073	0.1135	2.622	298.22
Na content	0.1094	0.1156	11.002	298.93
temperature reference	0.1081	0.1196	13.238	300.36
non-contributing variables omitted	0.1073	0.1121	4.658	281.02

TABLE C.8. BP1 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0737	0.0800	-	1610.03
target bath temperature	0.0936	0.1090	28.122	1490.27
bath temperature	0.1032	0.1042	22.915	1489.62
bath height	0.0764	0.0818	5.622	1488.40
bath resistivity	0.0748	0.0822	9.862	1490.79
emf	0.0700	0.0721	3.763	1489.61
AlF_3 addition (t-1)	0.0703	0.0758	3.073	1491.13
Na_2CO_3 addition (t-1)	0.0949	0.1102	31.864	1489.75
cell power	0.0939	0.1087	25.546	1490.43
cell age	0.0824	0.0831	14.610	1488.02
F content	0.0745	0.0813	3.910	1491.90
Na content	0.0736	0.0779	3.225	1490.54
temperature reference	0.0940	0.1012	18.507	1489.96
non-contributing variables omitted	0.0713	0.0774	4.065	1010.32

TABLE C.9. BP2 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0615	0.0719	-	3089.06
target bath temperature	0.0825	0.1017	14.289	2978.86
bath temperature	0.0794	0.0809	5.450	2982.05

bath height	0.0617	0.0811	6.495	2983.31
bath resistivity	0.0698	0.0854	10.210	2981.44
emf	0.0672	0.0803	5.406	2975.08
AlF_3 addition (t-1)	0.0730	0.0836	9.298	2982.59
Na_2CO_3 addition (t-1)	0.0646	0.0729	3.455	2976.73
cell power	0.0653	0.0778	4.342	2984.16
cell age	0.0674	0.0765	3.218	2983.91
F content	0.0687	0.0789	4.992	2979.14
Na content	0.0648	0.0793	4.971	2978.67
temperature reference	0.0777	0.0948	11.214	2981.52

TABLE C.10. RBF Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0635	0.0730	-	1457.08
target bath temperature	0.0693	0.0953	13.889	1384.34
bath temperature	0.0674	0.0942	11.303	1382.72
bath height	0.0638	0.0741	3.444	1385.62
bath resistivity	0.0660	0.0853	6.052	1384.70
emf	0.0630	0.0727	1.918	1385.91
AlF_3 addition (t-1)	0.0663	0.0861	6.005	1383.05
Na_2CO_3 addition (t-1)	0.0634	0.0729	1.255	1381.97
cell power	0.0641	0.0746	5.200	1384.41
cell age	0.0637	0.0732	2.382	1385.29
F content	0.0641	0.0738	3.279	1386.01
Na content	0.0638	0.0739	2.490	1384.38
temperature reference	0.0671	0.0937	8.379	1385.64
non-contributing variables omitted	0.0634	0.0727	1.232	1319.44

TABLE C.11. RBFKOH Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0607	0.0716	-	2955.95
target bath temperature	0.0672	0.0894	15.231	2878.78
bath temperature	0.0661	0.0871	11.115	2879.97
bath height	0.0607	0.0716	0.207	2878.16
bath resistivity	0.0648	0.0775	6.297	2876.71
emf	0.0604	0.0714	1.947	2881.66
AlF_3 addition (t-1)	0.0657	0.0800	9.661	2879.43
Na_2CO_3 addition (t-1)	0.0606	0.0715	1.051	2874.22
cell power	0.0648	0.0749	4.725	2876.81
cell age	0.0625	0.0726	2.668	2879.24
F content	0.0633	0.0724	2.621	2878.94
Na content	0.0606	0.0715	0.483	2876.92
temperature reference	0.0669	0.0886	12.802	2886.53
non-contributing variables omitted	0.0606	0.0715	1.671	2711.15

TABLE C.12. GRNN Predictive Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0549	0.0705	-	41.75
target bath temperature	0.0590	0.0802	7.174	38.83
bath temperature	0.0603	0.0731	3.014	37.94
bath height	0.0548	0.0697	2.582	37.81
bath resistivity	0.0592	0.0753	5.461	38.06
emf	0.0537	0.0692	3.158	38.15
AlF_3 addition (t-1)	0.0514	0.0754	6.305	37.73
Na_2CO_3 addition (t-1)	0.0548	0.0705	0.747	38.28
cell power	0.0562	0.0704	0.829	36.98
cell age	0.0548	0.0699	2.605	38.27
F content	0.0547	0.0686	4.893	39.06
Na content	0.0545	0.0692	3.350	37.94
temperature reference	0.0540	0.0732	3.127	38.81
non-contributing variables omitted	0.0548	0.0671	2.598	24.62

TABLE C.13. WH Network Casual Importance Results for Sigmoidal Activation Function in the Output Layer for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
target bath temperature	0.1085	0.1194	8.779	281.06
bath temperature	0.1128	0.1138	3.533	280.74
bath resistivity	0.1151	0.1289	18.272	281.38
AlF_3 addition (t-1)	0.1065	0.1129	3.261	282.09
Na_2CO_3 addition (t-1)	0.1067	0.1125	3.214	281.77
Na content	0.1088	0.1138	3.142	280.05
temperature reference	0.1083	0.1175	6.122	282.14

TABLE C.14. BP1 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
target bath temperature	0.0911	0.1067	18.214	1009.53
bath temperature	0.1005	0.1014	14.736	1011.42
bath height	0.0749	0.0791	3.851	1009.98
bath resistivity	0.0725	0.0798	5.145	1009.10
Na_2CO_3 addition (t-1)	0.0930	0.1083	22.233	1010.37
cell power	0.0921	0.1074	20.139	1009.63
cell age	0.0796	0.0802	6.767	1010.74
F content	0.0718	0.0784	3.636	1011.29
temperature reference	0.0918	0.0970	8.841	1009.75

TABLE C.15. BP2 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Additive Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
target bath temperature	0.0821	0.1009	13.764	3092.16
bath temperature	0.0804	0.0814	4.632	3094.63
bath height	0.0619	0.0812	6.595	3087.24
bath resistivity	0.0702	0.0861	11.068	3091.57
emf	0.0668	0.0798	5.827	3090.79

AlF_3 addition (t-1)	0.0727	0.0842	9.110	3091.48
Na_2CO_3 addition (t-1)	0.0653	0.0731	4.340	3089.04
cell power	0.0658	0.0768	4.812	3092.35
cell age	0.0681	0.0762	3.568	3088.46
F content	0.0684	0.0794	6.510	3089.12
Na content	0.0659	0.0806	4.435	3090.93
temperature reference	0.0768	0.0957	14.146	3092.55

TABLE C.16. RBF Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
target bath temperature	0.0691	0.0948	31.332	1321.50
bath temperature	0.0669	0.0944	29.851	1318.64
bath height	0.0637	0.0738	3.529	1319.02
bath resistivity	0.0658	0.0861	13.228	1322.61
AlF_3 addition (t-1)	0.0667	0.0863	13.746	1320.80
cell power	0.0638	0.0745	3.227	1319.15
cell age	0.0643	0.0730	2.419	1320.94
F content	0.0638	0.0735	3.015	1317.63
Na content	0.0642	0.0740	3.683	1321.81
temperature reference	0.0666	0.0931	22.384	1320.94

TABLE C.17. RBFKOH Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
target bath temperature	0.0641	0.0879	18.102	2712.44
bath temperature	0.0635	0.0864	13.909	2708.92
bath resistivity	0.0628	0.0768	7.359	2712.56
AlF_3 addition (t-1)	0.0624	0.0789	9.156	2712.69
cell power	0.0617	0.0742	5.325	2710.12
cell age	0.0614	0.0728	4.282	2711.96
F content	0.0616	0.0725	4.771	2709.59
temperature reference	0.0638	0.0861	14.643	2711.61

TABLE C.18. GRNN Casual Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Additive Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
target bath temperature	0.0590	0.0774	15.134	25.17
bath temperature	0.0603	0.0699	7.010	24.38
bath resistivity	0.0592	0.0721	9.096	25.15
AlF_3 addition (t-1)	0.0514	0.0729	10.431	24.52
temperature reference	0.0548	0.0698	5.318	24.87

TABLE C.19. Percentage Contribution of Input Variables using Predictive Importance Results for Electrolyte Additive Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
target bath temperature	19.7	20.5	22.9	23.7	25.5	39.3
bath temperature	5.8	17.1	6.9	22.5	22.2	10.5
bath height	0.0	1.3	7.1	1.2	0.0	0.0
bath resistivity	49.0	1.6	10.4	13.1	8.5	19.4

emf	0.0	0.0	6.4	0.0	0.0	0.0
AlF_3 addition (t-1)	2.9	0.0	9.0	13.9	12.1	19.8
Na_2CO_3 addition (t-1)	1.7	21.3	0.8	0.0	0.0	0.0
cell power	0.0	20.3	4.5	1.7	4.7	0.0
cell age	0.0	2.2	3.5	0.2	1.4	0.0
F content	0.0	0.9	5.4	0.8	1.1	0.0
Na content	4.6	0.0	5.7	1.0	0.0	0.0
temperature reference	16.2	15.0	17.6	22.0	24.4	10.9

TABLE C.20. Percentage Contribution of Input Variables using Casual Importance Results for Electrolyte Additive Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
target bath temperature	21.4	20.7	21.9	23.2	25.8	38.7
bath temperature	5.0	16.9	7.2	22.8	23.4	10.5
bath height	-	1.2	7.0	1.0	-	-
bath resistivity	49.3	1.7	10.7	14.0	8.3	18.8
emf addition	-	-	6.0	-	-	-
AlF_3 addition (t-1)	2.3	-	9.3	14.2	11.6	21.8
Na_2CO_3 addition (t-1)	1.2	21.8	0.9	-	-	-
cell power	-	21.2	3.7	1.7	4.2	-
cell age	-	2.0	3.2	0.1	2.0	-
F content	-	0.7	5.7	0.6	1.6	-
Na content	5.0	-	6.6	1.2	-	-
temperature reference	15.8	13.8	17.9	21.4	23.0	10.2

TABLE C.21. WH Network RMS Error and Computation Time Behaviour with Changing Network Architecture for Cell Failure Prediction Application

Output Layer Activation Function	RMS Error		Computation Time (s)
	Train	Test	
sigmoidal	0.0471	0.0773	311.82
linear	0.0887	0.1065	246.67

TABLE C.22. BP1 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application

Hidden Layer Nodes	RMS Error		Computation Time (s)
	Train	Test	
2	0.0066	0.0639	677.42
3	0.0032	0.0735	984.51
4	0.0023	0.0687	1261.32
5	0.0019	0.0652	1554.66
6	0.0019	0.0739	1876.80
7	0.0012	0.0719	2130.04
8	0.0017	0.0699	2437.62
9	0.0026	0.0646	2728.91
10	0.0021	0.0618	3026.15
12	0.0018	0.0743	3680.23
13	0.0016	0.0614	3997.62
14	0.0015	0.0591	4330.94
15	0.0014	0.0472	4570.06
16	0.0026	0.0638	4880.27
18	0.0036	0.0724	5340.03
20	0.0025	0.0643	5951.32

TABLE C.23. BP2 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application

Hidden Layer 1 Nodes	Hidden Layer 2 Nodes	RMS Error		Computation Time (s)
		Train	Test	
2	2	0.0694	0.0698	1747.33
2	3	0.0581	0.0693	1924.66
2	4	0.0504	0.0748	2132.18
2	5	0.0017	0.0728	2310.29
2	6	0.0031	0.0725	2481.31
2	7	0.0025	0.0764	2678.06
2	8	0.0012	0.0856	2867.05
2	9	0.0077	0.0902	3031.22
2	10	0.0261	0.0929	3165.38
3	2	0.0694	0.0699	1747.33
3	3	0.0128	0.0675	1924.66
3	4	0.0063	0.0750	2132.18
3	5	0.0072	0.0732	2310.29
3	6	0.0067	0.0729	2481.31
3	7	0.0030	0.0773	2678.06
3	8	0.0032	0.0860	2867.05
3	9	0.0019	0.0900	3031.22
3	10	0.0052	0.0928	3165.38
4	2	0.0693	0.0727	2118.70
4	3	0.0641	0.0679	2285.96
4	4	0.0655	0.0616	2461.01
4	5	0.0013	0.0627	2654.05
4	6	0.0003	0.0672	2851.65
4	7	0.0002	0.0618	3027.14
4	8	0.0012	0.0639	3187.78
4	9	0.0007	0.0682	3380.31
4	10	0.0009	0.0780	3525.62
5	2	0.0692	0.0695	2312.69
5	3	0.0018	0.0644	2482.48
5	4	0.0002	0.0596	2699.82
5	5	0.0003	0.0603	2891.73
5	6	0.0005	0.0628	3064.23
5	7	0.0006	0.0635	3236.06
5	8	0.0007	0.0660	3419.99
5	9	0.0009	0.0622	3601.13
5	10	0.0009	0.0615	3774.12
6	2	0.0692	0.0698	2528.28
6	3	0.0046	0.0615	2716.94
6	4	0.0005	0.0631	2905.87
6	5	0.0007	0.0581	3068.63
6	6	0.0005	0.0623	3230.42
6	7	0.0006	0.0616	3415.48
6	8	0.0013	0.0611	3611.49
6	9	0.0004	0.0627	3781.00
6	10	0.0008	0.0698	3967.78
7	2	0.0690	0.0693	2682.18
7	3	0.0051	0.0637	2850.98
7	4	0.0004	0.0628	3080.29
7	5	0.0008	0.0629	3268.02
7	6	0.0007	0.0627	3458.00
7	7	0.0005	0.0628	3649.59
7	8	0.0009	0.0643	3835.50
7	9	0.0011	0.0653	3974.03
7	10	0.0012	0.0709	4121.95

8	2	0.0654	0.0883	2912.14
8	3	0.0033	0.0740	3084.61
8	4	0.0005	0.0631	3275.07
8	5	0.0004	0.0636	3492.12
8	6	0.0006	0.0605	3658.41
8	7	0.0008	0.0638	3838.76
8	8	0.0006	0.0678	4004.62
8	9	0.0027	0.0722	4136.80
8	10	0.0036	0.0724	4300.41
9	2	0.0690	0.0692	3069.73
9	3	0.0650	0.0688	3255.96
9	4	0.0661	0.0707	3462.85
9	5	0.0011	0.0729	3654.35
9	6	0.0007	0.0808	3836.51
9	7	0.0006	0.0739	4000.85
9	8	0.0013	0.0686	4169.16
9	9	0.0015	0.0689	4371.58
9	10	0.0072	0.0718	4529.28
10	2	0.0715	0.0946	3271.22
10	3	0.0528	0.0748	3490.38
10	4	0.0052	0.0704	3674.52
10	5	0.0019	0.0610	3848.16
10	6	0.0003	0.0633	4053.02
10	7	0.0004	0.0666	4251.85
10	8	0.0017	0.0681	4443.65
10	9	0.0057	0.0743	4566.75
10	10	0.0098	0.0786	4720.21

TABLE C.24. RBF Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layer, Respectively, for Cell Failure Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1587	0.1702	274.57
0.1	10	0.1500	0.1658	544.79
0.1	20	0.1437	0.1646	1080.51
0.1	40	0.1254	0.1593	2087.72
0.1	60	0.1115	0.1650	3163.03
0.1	80	0.1195	0.1708	4138.26
0.1	100	0.1132	0.1766	5194.50
0.2	5	0.1494	0.1655	274.03
0.2	10	0.1395	0.1585	543.93
0.2	20	0.1386	0.1512	1081.86
0.2	40	0.1087	0.1509	2087.37
0.2	60	0.1162	0.1679	3162.78
0.2	80	0.1168	0.1656	4138.83
0.2	100	0.1182	0.1713	5193.41
0.3	5	0.1357	0.1558	275.02
0.3	10	0.1292	0.1555	544.24
0.3	20	0.1166	0.1437	1079.75
0.3	30	0.1166	0.1340	1574.24
0.3	40	0.1178	0.1287	2087.07
0.3	50	0.1189	0.1346	2633.77
0.3	60	0.1139	0.1462	3163.70
0.3	80	0.1159	0.1562	4137.46
0.3	100	0.1218	0.1604	5195.19
0.4	5	0.1404	0.1574	274.63
0.4	10	0.1245	0.1516	543.78

0.4	20	0.1143	0.1433	1079.96
0.4	40	0.1255	0.1363	2087.74
0.4	60	0.1192	0.1518	3164.29
0.4	80	0.1185	0.1547	4137.97
0.4	100	0.1249	0.1502	5194.21
0.5	5	0.1351	0.1588	273.93
0.5	10	0.1237	0.1498	543.82
0.5	20	0.1171	0.1459	1080.62
0.5	40	0.1188	0.1413	2088.03
0.5	60	0.1219	0.1511	3164.59
0.5	80	0.1196	0.1488	4138.38
0.5	100	0.1168	0.1578	5193.67
0.6	5	0.1346	0.1541	274.02
0.6	10	0.1200	0.1505	544.93
0.6	20	0.1109	0.1445	1081.17
0.6	40	0.1151	0.1455	2087.77
0.6	60	0.1207	0.1499	3164.28
0.6	80	0.1192	0.1533	4137.84
0.6	100	0.1241	0.1556	5194.33
0.7	5	0.1439	0.1585	273.75
0.7	10	0.1290	0.1586	543.73
0.7	20	0.1213	0.1456	1081.48
0.7	40	0.1133	0.1442	2087.93
0.7	60	0.1208	0.1477	3164.87
0.7	80	0.1152	0.1535	4138.41
0.7	100	0.1220	0.1592	5193.71
0.9	5	0.1393	0.1592	273.98
0.9	10	0.1290	0.1599	543.24
0.9	20	0.1246	0.1454	1079.99
0.9	40	0.1263	0.1480	2087.90
0.9	60	0.1309	0.1509	3163.72
0.9	80	0.1336	0.1541	4138.67
0.9	100	0.1345	0.1619	5193.80

TABLE C.25. RBFKOH Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layer, Respectively, for Cell Failure Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.1795	0.2005	1980.89
0.1	10	0.1975	0.1979	2458.99
0.1	20	0.1487	0.1727	3437.93
0.1	40	0.1305	0.1472	5158.00
0.1	60	0.1310	0.1483	6962.27
0.1	80	0.1323	0.1467	8673.35
0.1	100	0.1415	0.1602	10510.89
0.2	5	0.1851	0.1895	1980.30
0.2	10	0.1931	0.2052	2459.55
0.2	20	0.1250	0.1522	3437.36
0.2	40	0.1017	0.1251	5157.40
0.2	60	0.1012	0.1299	6961.74
0.2	80	0.1084	0.1284	8673.20
0.2	100	0.1023	0.1391	10510.76
0.3	5	0.1761	0.1933	1980.65
0.3	10	0.1809	0.1917	2459.32
0.3	20	0.1220	0.1549	3437.74
0.3	40	0.0970	0.1158	5158.17
0.3	60	0.1013	0.1275	6962.23

0.3	80	0.1055	0.1259	8672.74
0.3	100	0.1062	0.1276	10511.88
0.4	5	0.1780	0.1821	1981.01
0.4	10	0.1614	0.1688	2459.29
0.4	20	0.1226	0.1451	3437.00
0.4	30	0.1134	0.1381	4292.08
0.4	40	0.1043	0.1081	5157.87
0.4	50	0.1081	0.1137	6094.02
0.4	60	0.1113	0.1160	6961.36
0.4	80	0.1108	0.1199	8673.24
0.4	100	0.1187	0.1235	10511.73
0.5	5	0.1621	0.1710	1980.72
0.5	10	0.1458	0.1615	2459.97
0.5	20	0.1203	0.1474	3438.09
0.5	40	0.1122	0.1190	5158.45
0.5	60	0.1099	0.1271	6961.56
0.5	80	0.1258	0.1290	8673.42
0.5	100	0.1284	0.1370	10512.10
0.6	5	0.1656	0.1685	1980.32
0.6	10	0.1391	0.1507	2459.40
0.6	20	0.1143	0.1459	3437.61
0.6	40	0.1179	0.1367	5157.58
0.6	60	0.1272	0.1384	6961.16
0.6	80	0.1228	0.1389	8672.95
0.6	100	0.1278	0.1427	10510.37
0.7	5	0.1475	0.1649	1981.13
0.7	10	0.1404	0.1468	2459.03
0.7	20	0.1125	0.1384	3438.01
0.7	40	0.1220	0.1305	5158.34
0.7	60	0.1219	0.1356	6962.06
0.7	80	0.1254	0.1327	8672.57
0.7	100	0.1301	0.1448	10511.06
0.9	5	0.1399	0.1453	1981.22
0.9	10	0.1340	0.1374	2459.45
0.9	20	0.1119	0.1309	3438.26
0.9	40	0.1269	0.1294	5157.49
0.9	60	0.1371	0.1381	6961.61
0.9	80	0.1379	0.1382	8673.55
0.9	100	0.1417	0.1463	10510.37

TABLE C.26. GRNN RMS Error and Computation Time Behaviour with Changing Network Architecture using an Exponential and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Cell Failure Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	10	0.2089	0.2265	1.65
0.1	50	0.2020	0.2256	5.71
0.1	100	0.1871	0.2120	9.83
0.1	200	0.1524	0.1838	17.79
0.1	300	0.1335	0.1764	26.19
0.1	400	0.1326	0.1620	34.89
0.1	500	0.1225	0.1794	41.81
0.1	600	0.1091	0.1744	49.63
0.1	800	0.0995	0.1890	66.09
0.1	1000	0.0898	0.1959	81.95
0.1	1500	0.0815	0.1907	122.10
0.2	10	0.1995	0.2026	1.65
0.2	50	0.1875	0.2025	5.73

0.2	100	0.1784	0.1982	9.80
0.2	200	0.1550	0.1897	17.81
0.2	400	0.1435	0.1779	34.85
0.2	600	0.1391	0.1867	49.59
0.2	800	0.1279	0.1823	66.05
0.2	1000	0.1040	0.1952	81.94
0.2	1500	0.0834	0.1837	122.08
0.3	10	0.1849	0.2070	1.66
0.3	50	0.1886	0.1955	5.73
0.3	100	0.1905	0.1915	9.81
0.3	200	0.1605	0.1886	17.82
0.3	400	0.1558	0.1863	34.89
0.3	600	0.1525	0.1899	49.62
0.3	800	0.1626	0.1739	66.04
0.3	1000	0.1558	0.1894	81.90
0.3	1500	0.1226	0.1933	122.12
0.5	10	0.1958	0.1977	1.68
0.5	50	0.1895	0.2017	5.74
0.5	100	0.1845	0.2007	9.81
0.5	200	0.1731	0.1908	17.79
0.5	400	0.1810	0.1981	34.86
0.5	600	0.1820	0.1942	49.62
0.5	800	0.1800	0.1976	66.10
0.5	1000	0.1711	0.1972	89.92
0.5	1500	0.1663	0.2000	122.11
0.7	10	0.1999	0.2041	1.65
0.7	50	0.1932	0.2050	5.72
0.7	100	0.1984	0.1988	9.79
0.7	200	0.1768	0.2008	17.78
0.7	400	0.1902	0.1914	34.87
0.7	600	0.1842	0.1907	49.60
0.7	800	0.1838	0.2001	66.08
0.7	1000	0.1750	0.2050	89.93
0.7	1500	0.1755	0.2008	122.12
0.9	10	0.1920	0.2021	1.64
0.9	50	0.2028	0.2051	5.70
0.9	100	0.2014	0.2035	9.82
0.9	200	0.1808	0.1875	17.76
0.9	400	0.1810	0.1899	34.89
0.9	600	0.1953	0.1984	49.59
0.9	800	0.1758	0.1968	66.06
0.9	1000	0.1828	0.1973	89.90
0.9	1500	0.1888	0.1946	122.15

TABLE C.27. WH Network Predictive Importance Results for Sigmoidal Activation Function in the Output Layer for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0471	0.0773	-	331.82
Fe weekly	0.0539	0.0783	3.168	302.40
Si weekly	0.0473	0.0777	2.475	302.38
lining voltage drop	0.0541	0.0885	11.648	302.52
bath height	0.0467	0.0771	1.738	302.76
bath temperature	0.0476	0.0779	3.261	302.93
cell age	0.0485	0.0781	3.730	301.86
AE frequency	0.0505	0.0860	9.503	302.85
AE duration	0.0489	0.0786	4.297	301.93
unscheduled anode change	0.0732	0.0866	9.843	302.50

rod height	0.0521	0.0783	3.133	301.97
cell power	0.0470	0.0776	2.405	301.90
high frequency noise	0.0482	0.0791	6.423	303.04
low frequency noise	0.0471	0.0771	1.929	302.42
cell resistance	0.0499	0.0786	4.350	302.74
emf	0.0470	0.0746	3.962	301.87
bath resistivity	0.0471	0.0780	3.425	301.99
cell voltage	0.0693	0.0803	8.745	302.37
AE energy	0.0479	0.0805	7.756	302.86
Si	0.0477	0.0778	2.546	302.64
Fe	0.0468	0.0778	3.457	301.96
Fe/V	0.0472	0.0781	2.833	303.11
Fe/Ga	0.0463	0.0770	3.118	302.44
high temperature count	0.0827	0.1032	22.484	301.89
low temperature count	0.0485	0.0783	4.187	302.05
non-contributing variables omitted	0.0470	0.0761	3.862	271.28

TABLE C.28. BP1 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application (t-critical = 2.333, α = 0.01, df = 499)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0014	0.0472	-	4570.06
Fe weekly	0.0051	0.0634	6.008	4508.01
Si weekly	0.0025	0.0659	7.510	4506.82
lining voltage drop	0.0037	0.0668	7.915	4507.68
bath height	0.0029	0.0665	7.970	4509.44
bath temperature	0.0033	0.0635	5.670	4508.46
cell age	0.0022	0.0726	14.780	4506.76
AE frequency	0.0054	0.0638	6.819	4508.87
AE duration	0.0027	0.0632	6.436	4507.64
unscheduled anode change	0.0030	0.0774	19.557	4507.92
rod height	0.0022	0.0635	6.609	4509.78
cell power	0.0041	0.0753	18.465	4507.40
high frequency noise	0.0024	0.0718	12.371	4508.90
low frequency noise	0.0043	0.0633	5.011	4508.46
cell resistance	0.0029	0.0661	7.510	4509.23
emf	0.0036	0.0511	5.237	4509.65
bath resistivity	0.0031	0.0511	5.911	4507.54
cell voltage	0.0037	0.0505	4.225	4509.86
AE energy	0.0036	0.0638	6.459	4509.72
Si	0.0014	0.0634	5.845	4507.83
Fe	0.0028	0.0721	12.174	4508.34
Fe/V	0.0043	0.0683	8.785	4506.08
Fe/Ga	0.0029	0.0686	8.671	4507.92
high temperature count	0.0048	0.0750	18.395	4509.14
low temperature count	0.0021	0.0451	4.181	4507.61
non-contributing variables omitted	0.0021	0.0451	4.698	4438.21

TABLE C.29. BP2 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application (t-critical = 2.333, α = 0.01, df = 499)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0007	0.0581	-	3068.63
Fe weekly	0.0011	0.0681	7.601	2993.16
Si weekly	0.0011	0.0684	7.804	2993.17
lining voltage drop	0.0023	0.0825	11.112	2994.42

bath height	0.0007	0.0579	0.504	2994.16
bath temperature	0.0009	0.0593	2.625	2993.31
cell age	0.0014	0.0799	9.527	2994.41
AE frequency	0.0017	0.0842	14.323	2994.02
AE duration	0.0012	0.0664	7.057	2994.54
unscheduled anode change	0.0021	0.0811	10.370	2993.59
rod height	0.0017	0.0699	8.331	2994.11
cell power	0.0015	0.0712	8.034	2994.40
high frequency noise	0.0010	0.0594	2.688	2993.69
low frequency noise	0.0009	0.0602	3.501	2994.50
cell resistance	0.0013	0.0715	8.682	2993.64
emf	0.0006	0.0580	0.340	2994.57
bath resistivity	0.0009	0.0599	3.460	2994.02
cell voltage	0.0015	0.0732	8.366	2994.22
AE energy	0.0018	0.0782	9.798	2993.66
Si	0.0010	0.0627	4.924	2994.19
Fe	0.0012	0.0631	5.171	2993.84
Fe/V	0.0011	0.0634	5.696	2993.25
Fe/Ga	0.0010	0.0629	4.721	2994.13
high temperature count	0.0024	0.0819	11.935	2993.44
low temperature count	0.0008	0.0613	4.364	2993.22
non-contributing variables omitted	0.0007	0.0580	0.350	2871.05

TABLE C.30. RBF Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.1178	0.1287	-	2087.07
Fe weekly	0.1261	0.1444	9.642	2001.69
Si weekly	0.1233	0.1470	11.719	1999.24
lining voltage drop	0.1259	0.1407	7.338	2001.20
bath height	0.1103	0.1274	3.270	2000.07
bath temperature	0.1253	0.1387	4.229	2000.77
cell age	0.1241	0.1443	9.961	2001.18
AE frequency	0.1258	0.1394	5.851	2001.15
AE duration	0.1244	0.1413	8.418	2002.39
unscheduled anode change	0.1253	0.1484	12.826	2000.88
rod height	0.1216	0.1386	4.201	2002.37
cell power	0.1273	0.1527	15.956	2001.00
high frequency noise	0.1294	0.1501	14.296	2001.84
low frequency noise	0.1217	0.1421	9.642	2002.31
cell resistance	0.1256	0.1383	4.349	2001.81
emf	0.1212	0.1229	3.854	2002.06
bath resistivity	0.1144	0.1278	2.751	2001.48
cell voltage	0.1248	0.1232	3.252	2001.71
AE energy	0.1237	0.1371	4.812	2000.37
Si	0.1165	0.1218	4.170	2002.55
Fe	0.1252	0.1493	13.563	2001.85
Fe/V	0.1180	0.1386	4.027	2001.98
Fe/Ga	0.1266	0.1467	11.848	2002.49
high temperature count	0.1215	0.1560	21.918	2001.05
low temperature count	0.1140	0.1253	2.906	2002.38
non-contributing variables omitted	0.1156	0.1279	3.543	1793.26

TABLE C.31. RBFKOH Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.1043	0.1081	-	5157.87
Fe weekly	0.1109	0.1225	11.432	5082.34
Si weekly	0.1039	0.1079	2.507	5083.09
lining voltage drop	0.1084	0.1190	8.006	5083.64
bath height	0.0926	0.1021	3.573	5082.71
bath temperature	0.1086	0.1139	4.241	5084.31
cell age	0.1089	0.1207	9.898	5083.25
AE frequency	0.1080	0.1172	6.240	5083.29
AE duration	0.1040	0.1080	0.257	5083.02
unscheduled anode change	0.1088	0.1271	13.835	5083.37
rod height	0.1143	0.1161	5.968	5083.86
cell power	0.1126	0.1314	19.977	5082.36
high frequency noise	0.1121	0.1280	13.628	5083.79
low frequency noise	0.1061	0.1171	6.484	5083.68
cell resistance	0.1040	0.1078	2.824	5083.57
emf	0.1032	0.0999	3.745	5082.97
bath resistivity	0.0992	0.1045	2.930	5082.63
cell voltage	0.1041	0.0993	3.788	5082.81
AE energy	0.1072	0.1142	5.302	5084.30
Si	0.0972	0.0977	3.865	5082.86
Fe	0.1106	0.1244	12.620	5084.27
Fe/V	0.1046	0.1146	5.269	5084.04
Fe/Ga	0.1038	0.1081	0.146	5084.08
high temperature count	0.1062	0.1307	17.141	5084.17
low temperature count	0.0998	0.1001	3.956	5082.70
non-contributing variables omitted	0.1031	0.1074	2.547	4547.39

TABLE C.32. GRNN Predictive Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.1326	0.1620	-	34.89
Fe weekly	0.1322	0.1626	2.798	33.96
Si weekly	0.1325	0.1630	3.607	33.98
lining voltage drop	0.1457	0.1750	15.272	33.97
bath height	0.1317	0.1622	2.573	33.94
bath temperature	0.1307	0.1612	3.887	33.96
cell age	0.1304	0.1611	3.987	33.95
AE frequency	0.1291	0.1596	4.088	33.99
AE duration	0.1278	0.1579	5.804	33.95
unscheduled anode change	0.1267	0.1569	6.915	33.96
rod height	0.1286	0.1589	5.504	33.96
cell power	0.1305	0.1611	3.334	33.97
high frequency noise	0.1335	0.1641	4.205	33.95
low frequency noise	0.1361	0.1672	7.241	33.98
cell resistance	0.1315	0.1614	3.580	33.97
emf	0.1303	0.1613	3.918	33.95
bath resistivity	0.1310	0.1615	3.881	33.96
cell voltage	0.1321	0.1627	3.159	33.96
AE energy	0.1300	0.1602	4.299	33.95
Si	0.1325	0.1630	3.549	33.97
Fe	0.1319	0.1623	3.313	33.98
Fe/V	0.1325	0.1630	3.939	33.97

Fe/Ga	0.1316	0.1620	2.307	33.98
high temperature count	0.1327	0.1629	3.438	33.97
low temperature count	0.1320	0.1623	2.643	33.98
non-contributing variables omitted	0.1319	0.1598	4.595	24.84

TABLE C.33. WH Network Casual Importance Results for Sigmoidal Activation Function in the Output Layer for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
Fe weekly	0.0540	0.0770	3.104	270.84
Si weekly	0.0471	0.0762	2.568	271.55
lining voltage drop	0.0542	0.0877	10.495	270.64
bath temperature	0.0474	0.0768	3.328	271.48
cell age	0.0483	0.0769	3.664	271.92
AE frequency	0.0507	0.0847	8.464	271.50
AE duration	0.0492	0.0775	4.379	272.07
unscheduled anode change	0.0768	0.0854	8.860	272.31
rod height	0.0520	0.0774	4.069	271.73
cell power	0.0476	0.0768	3.584	270.82
high frequency noise	0.0479	0.0778	4.327	271.08
cell resistance	0.0504	0.0773	3.834	270.99
bath resistivity	0.0468	0.0767	3.830	272.15
cell voltage	0.0687	0.0790	4.006	271.84
AE energy	0.0482	0.0791	5.264	271.91
Si	0.0477	0.0766	3.329	271.96
Fe	0.0469	0.0769	3.110	272.45
Fe/V	0.0481	0.0772	3.597	270.67
high temperature count	0.0832	0.1020	17.056	271.88
low temperature count	0.0481	0.0771	3.327	271.39

TABLE C.34. BP1 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
Fe weekly	0.0049	0.0632	9.558	4437.15
Si weekly	0.0025	0.0659	11.822	4438.09
lining voltage drop	0.0034	0.0667	13.024	4439.72
bath height	0.0027	0.0665	13.294	4438.64
bath temperature	0.0032	0.0635	9.822	4437.08
cell age	0.0021	0.0725	19.857	4439.24
AE frequency	0.0053	0.0637	9.854	4438.73
AE duration	0.0026	0.0632	9.498	4437.55
unscheduled anode change	0.0028	0.0774	27.933	4437.50
rod height	0.0021	0.0634	9.942	4438.81
cell power	0.0040	0.0752	22.712	4438.49
high frequency noise	0.0023	0.0715	18.592	4439.62
low frequency noise	0.0042	0.0630	9.886	4438.27
cell resistance	0.0027	0.0659	11.285	4437.43
emf	0.0036	0.0510	7.466	4438.80
bath resistivity	0.0029	0.0510	7.751	4439.37
cell voltage	0.0037	0.0503	5.710	4439.07
AE energy	0.0034	0.0637	9.417	4437.00
Si	0.0012	0.0632	9.465	4438.94
Fe	0.0028	0.0718	18.203	4437.52
Fe/V	0.0040	0.0682	15.299	4438.51
Fe/Ga	0.0026	0.0685	15.921	4438.90
high temperature count	0.0048	0.0749	22.960	4437.25

TABLE C.35. BP2 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Cell Failure Prediction Application (t-critical = 2.333, α = 0.01, df = 499)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
Fe weekly	0.0010	0.0681	7.298	2871.15
Si weekly	0.0011	0.0676	7.082	2871.70
lining voltage drop	0.0023	0.0818	15.371	2871.15
bath temperature	0.0009	0.0586	2.586	2872.44
cell age	0.0013	0.0794	12.993	2871.68
AE frequency	0.0016	0.0833	19.589	2872.13
AE duration	0.0011	0.0657	5.186	2872.17
unscheduled anode change	0.0021	0.0806	12.322	2872.12
rod height	0.0016	0.0698	8.473	2871.58
cell power	0.0015	0.0703	9.018	2871.76
high frequency noise	0.0010	0.0584	2.500	2871.73
low frequency noise	0.0008	0.0600	3.117	2872.25
cell resistance	0.0012	0.0706	9.070	2872.67
bath resistivity	0.0009	0.0593	3.527	2872.45
cell voltage	0.0015	0.0725	11.778	2871.96
AE energy	0.0017	0.0777	12.266	2872.81
Si	0.0009	0.0624	4.031	2872.34
Fe	0.0012	0.0626	4.639	2871.16
Fe/V	0.0011	0.0630	4.488	2871.47
Fe/Ga	0.0009	0.0619	4.787	2872.22
high temperature count	0.0023	0.0814	15.074	2872.70
low temperature count	0.0008	0.0608	3.044	2872.53

TABLE C.36. RBF Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, α = 0.01, df = 499)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
Fe weekly	0.1260	0.1439	10.341	1792.85
Si weekly	0.1213	0.1466	13.523	1791.30
lining voltage drop	0.1236	0.1406	8.908	1791.27
bath temperature	0.1241	0.1368	3.750	1791.03
cell age	0.1233	0.1442	10.708	1793.89
AE frequency	0.1256	0.1385	5.692	1792.55
AE duration	0.1229	0.1398	7.459	1793.20
unscheduled anode change	0.1237	0.1473	13.692	1791.38
rod height	0.1203	0.1361	3.238	1792.57
cell power	0.1263	0.1513	21.097	1792.08
high frequency noise	0.1183	0.1427	9.550	1792.07
low frequency noise	0.1199	0.1420	9.280	1791.60
cell resistance	0.1254	0.1373	4.588	1792.56
AE energy	0.1214	0.1369	3.683	1792.37
Fe	0.1228	0.1486	15.047	1792.92
Fe/V	0.1163	0.1383	5.921	1791.84
Fe/Ga	0.1184	0.1399	7.428	1793.05
high temperature count	0.1204	0.1537	25.411	1792.51

TABLE C.37. RBFKOH Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, α = 0.01, df = 499)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
Fe weekly	0.1073	0.1204	12.451	4548.14
lining voltage drop	0.1043	0.1178	8.544	4549.24

bath temperature	0.1048	0.1091	4.432	4549.49
cell age	0.1073	0.1160	8.721	4550.63
AE frequency	0.1095	0.1120	6.050	4548.77
unscheduled anode change	0.1063	0.1201	12.060	4550.21
rod height	0.1058	0.1151	8.718	4549.04
cell power	0.1101	0.1220	14.431	4550.03
high frequency noise	0.1053	0.1214	12.882	4548.34
low frequency noise	0.1045	0.1112	6.638	4549.36
AE energy	0.1063	0.1099	4.872	4550.03
Fe	0.1068	0.1188	9.636	4548.89
Fe/V	0.1074	0.1112	6.511	4550.58
high temperature count	0.1048	0.1239	17.442	4549.85

TABLE C.38. GRNN Casual Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Cell Failure Prediction Application (t-critical = 2.333, $\alpha = 0.01$, $df = 499$)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
Fe weekly	0.1330	0.1605	3.139	24.83
Si weekly	0.1333	0.1609	3.762	24.86
lining voltage drop	0.1465	0.1727	13.093	24.85
bath height	0.1319	0.1600	3.314	24.82
high frequency noise	0.1339	0.1621	5.960	24.84
low frequency noise	0.1372	0.1653	8.423	24.86
cell voltage	0.1330	0.1607	3.511	24.85
Si	0.1319	0.1619	5.487	24.86
Fe	0.1324	0.1599	2.409	24.82
Fe/V	0.1325	0.1608	3.159	24.85
high temperature count	0.1338	0.1606	3.311	24.86
low temperature count	0.1324	0.1602	3.264	24.85

TABLE C.39. Percentage Contribution of Input Variables using Predictive Importance Results for Cell Failure Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
Fe weekly	1.4	3.8	4.0	5.7	7.8	2.3
Si weekly	0.5	4.4	4.1	6.6	0.0	3.8
lining voltage drop	15.3	4.7	9.7	4.3	5.9	49.4
bath height	0.0	4.6	0.0	0.0	0.0	0.8
bath temperature	0.8	3.9	0.5	3.6	3.2	0.0
cell age	1.1	6.0	8.7	5.6	6.9	0.0
AE frequency	11.9	3.9	10.4	3.9	5.0	0.0
AE duration	1.8	3.8	3.3	4.5	0.0	0.0
unsched. anode change	12.7	7.2	9.2	7.1	10.4	0.0
rod height	1.4	3.9	4.7	3.6	4.4	0.0
cell power	0.4	6.7	5.2	8.7	12.7	0.0
high frequency noise	2.5	5.8	0.5	7.7	10.8	8.0
low frequency noise	0.0	3.8	0.8	4.8	4.9	19.8
cell resistance	1.8	4.5	5.3	3.5	0.0	0.0
emf	0.0	0.9	0.0	0.0	0.0	0.0
bath resistivity	1.0	0.9	0.7	0.0	0.0	0.0
cell voltage	4.1	0.8	6.0	0.0	0.0	2.7
AE energy	4.4	3.9	8.0	3.0	3.3	0.0
Si content	0.7	3.8	1.8	0.0	0.0	3.8
Fe content	0.7	5.9	2.0	7.4	8.9	1.1
Fe/V	1.1	5.0	2.1	3.6	3.5	3.8
Fe/Ga	0.0	5.1	1.9	6.5	0.0	0.0
high temperature count	35.3	6.6	9.5	9.9	12.3	3.4
low temperature count	1.4	0.0	1.3	0.0	0.0	1.1

TABLE C.40. Percentage Contribution of Input Variables using Casual Importance Results for Cell Failure Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
Fe weekly	1.2	3.9	4.2	6.1	10.4	2.5
Si weekly	0.1	4.5	4.0	7.1	-	3.9
lining voltage drop	15.7	4.6	9.9	4.8	8.3	46.1
bath height	-	4.6	-	-	-	0.7
bath temperature	0.9	3.9	0.3	3.4	1.4	-
cell age	1.1	5.9	8.9	6.2	6.9	-
AE frequency	11.6	4.0	10.6	4.0	3.7	-
AE duration	1.9	3.9	3.2	4.5	-	-
unsched. anode change	12.6	6.9	9.4	7.4	10.1	-
rod height	1.8	3.9	4.9	3.1	6.1	-
cell power	0.9	6.4	5.1	8.9	11.7	-
high frequency noise	2.3	5.7	0.2	5.6	11.2	8.2
low frequency noise	-	3.8	0.8	5.4	3.0	19.6
cell resistance	1.6	4.5	5.3	3.6	-	-
emf	-	1.3	-	-	-	-
bath resistivity	0.8	1.3	0.5	-	-	-
cell voltage	3.9	1.1	6.0	-	-	3.2
AE energy	4.0	4.0	8.2	3.4	2.0	-
Si content	0.7	3.9	1.8	-	-	7.5
Fe content	1.1	5.7	1.9	7.9	9.1	0.4
Fe/V	1.5	4.9	2.1	4.0	3.0	3.6
Fe/Ga	-	5.0	1.6	4.6	-	-
high temperature count	35.0	6.4	9.8	9.8	13.2	2.9
low temperature count	1.3	-	1.2	-	-	1.4

TABLE C.41. WH Network RMS Error and Computation Time Behaviour with Changing Network Architecture for Electrolyte Temperature Prediction Application

Output Layer	RMS Error		Computation
Activation Function	Train	Test	Time (s)
sigmoidal	0.0623	0.0627	184.67
linear	0.0629	0.0635	116.06

TABLE C.42. BP1 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application

Hidden Layer	RMS Error		Computation
Nodes	Train	Test	Time (s)
2	0.0621	0.0637	650.23
3	0.0619	0.0635	764.79
4	0.0620	0.0633	876.12
5	0.0620	0.0632	1016.92
6	0.0621	0.0632	1150.38
7	0.0623	0.0634	1250.14
8	0.0619	0.0639	1372.61
9	0.0620	0.0635	1481.04
10	0.0619	0.0635	1606.47
15	0.0624	0.0638	2102.41
20	0.0682	0.0682	2623.73

TABLE C.43. BP2 Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application

Hidden Layer 1 Nodes	Hidden Layer 2 Nodes	RMS Error		Computation Time (s)
		Train	Test	
2	2	0.0627	0.0653	738.29
2	3	0.0628	0.0658	920.21
2	4	0.0611	0.0654	1120.58
2	5	0.0615	0.0662	1299.67
2	6	0.0635	0.0661	1471.17
2	7	0.0633	0.0659	1668.38
2	8	0.0632	0.0662	1856.91
2	9	0.0634	0.0660	2025.53
2	10	0.0632	0.0673	2159.08
3	2	0.0627	0.0652	922.34
3	3	0.0600	0.0627	1076.67
3	4	0.0622	0.0644	1277.73
3	5	0.0623	0.0649	1451.07
3	6	0.0622	0.0652	1618.38
3	7	0.0621	0.0664	1805.06
3	8	0.0632	0.0675	1985.46
3	9	0.0630	0.0678	2112.72
3	10	0.0629	0.0681	2259.90
4	2	0.0585	0.0641	1107.26
4	3	0.0604	0.0628	1275.82
4	4	0.0602	0.0633	1451.45
4	5	0.0629	0.0657	1644.37
4	6	0.0595	0.0620	1845.96
4	7	0.0612	0.0633	2021.34
4	8	0.0627	0.0661	2182.09
4	9	0.0632	0.0655	2374.01
4	10	0.0633	0.0669	2519.93
5	2	0.0598	0.0634	1307.00
5	3	0.0594	0.0626	1476.68
5	4	0.0590	0.0634	1693.52
5	5	0.0595	0.0613	1885.43
5	6	0.0603	0.0645	2053.56
5	7	0.0606	0.0651	2230.37
5	8	0.0602	0.0643	2414.11
5	9	0.0604	0.0642	2591.45
5	10	0.0606	0.0645	2767.82
6	2	0.0595	0.0638	1518.14
6	3	0.0590	0.0646	1706.27
6	4	0.0596	0.0596	1895.73
6	5	0.0568	0.0580	2056.69
6	6	0.0568	0.0583	2224.62
6	7	0.0580	0.0618	2409.18
6	8	0.0584	0.0620	2601.93
6	9	0.0589	0.0641	2770.86
6	10	0.0591	0.0642	2957.64
7	2	0.0584	0.0611	1676.38
7	3	0.0574	0.0616	1845.29
7	4	0.0570	0.0610	2070.73
7	5	0.0626	0.0633	2262.14
7	6	0.0646	0.0635	2447.33
7	7	0.0651	0.0637	2639.91
7	8	0.0649	0.0641	2824.06
7	9	0.0652	0.0645	2967.73
7	10	0.0654	0.0651	3116.15

8	2	0.0576	0.0624	1900.20
8	3	0.0595	0.0635	2078.81
8	4	0.0573	0.0595	2265.39
8	5	0.0587	0.0597	2480.18
8	6	0.0593	0.0608	2648.27
8	7	0.0601	0.0615	2826.82
8	8	0.0602	0.0624	2995.06
8	9	0.0609	0.0621	3126.66
8	10	0.0611	0.0631	3294.53
9	2	0.0569	0.0600	2059.59
9	3	0.0576	0.0622	2246.28
9	4	0.0575	0.0638	2452.71
9	5	0.0581	0.0621	2648.47
9	6	0.0611	0.0634	2826.37
9	7	0.0612	0.0635	2995.16
9	8	0.0619	0.0639	3163.28
9	9	0.0621	0.0649	3360.14
9	10	0.0623	0.0654	3519.60
10	2	0.0566	0.0605	2259.28
10	3	0.0553	0.0573	2480.24
10	4	0.0565	0.0599	2668.83
10	5	0.0577	0.0606	2836.72
10	6	0.0586	0.0623	3041.08
10	7	0.0567	0.0610	3246.05
10	8	0.0573	0.0612	3433.97
10	9	0.0579	0.0617	3556.61
10	10	0.0584	0.0623	3714.52
11	2	0.0588	0.0622	2531.23
11	3	0.0560	0.0611	2776.68
11	4	0.0572	0.0626	2964.42
11	5	0.0597	0.0630	3143.05
11	6	0.0601	0.0631	3319.61
11	7	0.0595	0.0634	3522.78
11	8	0.0606	0.0637	3716.93
11	9	0.0609	0.0642	3904.62
11	10	0.0615	0.0649	4053.14
12	2	0.0572	0.0632	2864.32
12	3	0.0571	0.0613	3090.14
12	4	0.0594	0.0632	3321.46
12	5	0.0611	0.0645	3527.72
12	6	0.0609	0.0648	3688.84
12	7	0.0607	0.0645	3860.99
12	8	0.0614	0.0649	4032.37
12	9	0.0617	0.0651	4216.21
12	10	0.0615	0.0652	4422.06
13	2	0.0563	0.0610	3204.94
13	3	0.0565	0.0611	3461.71
13	4	0.0560	0.0611	3658.62
13	5	0.0568	0.0603	3892.57
13	6	0.0571	0.0612	4056.83
13	7	0.0578	0.0617	4200.02
13	8	0.0582	0.0624	4398.64
13	9	0.0588	0.0627	4568.53
13	10	0.0591	0.0634	4751.29
14	2	0.0578	0.0635	3561.17
14	3	0.0601	0.0629	3784.74
14	4	0.0574	0.0618	4008.63
14	5	0.0597	0.0624	4244.46
14	6	0.0604	0.0623	4396.29

14	7	0.0598	0.0628	4527.54
14	8	0.0599	0.0631	4701.83
14	9	0.0612	0.0642	4893.47
14	10	0.0607	0.0638	5076.92
15	2	0.0586	0.0625	3952.40
15	3	0.0594	0.0617	4163.73
15	4	0.0588	0.0622	4407.67
15	5	0.0593	0.0634	4591.99
15	6	0.0601	0.0629	4777.34
15	7	0.0597	0.0631	4902.06
15	8	0.0604	0.0627	5091.55
15	9	0.0609	0.0635	5296.10
15	10	0.0612	0.0643	5468.03

TABLE C.44. RBF Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.0791	0.0882	156.42
0.1	10	0.0703	0.0816	304.19
0.1	20	0.0666	0.0775	605.63
0.1	40	0.0651	0.0737	1220.25
0.1	60	0.0647	0.0730	1796.07
0.1	80	0.0654	0.0741	2388.25
0.1	100	0.0669	0.0752	2952.05
0.2	5	0.0770	0.0874	157.09
0.2	10	0.0685	0.0795	303.97
0.2	20	0.0652	0.0761	606.12
0.2	30	0.0646	0.0705	945.94
0.2	40	0.0631	0.0713	1220.22
0.2	50	0.0637	0.0712	1572.29
0.2	60	0.0651	0.0719	1796.94
0.2	80	0.0661	0.0729	2384.27
0.2	100	0.0672	0.0758	2951.14
0.3	5	0.0724	0.0827	156.37
0.3	10	0.0674	0.0771	304.35
0.3	20	0.0645	0.0723	605.88
0.3	30	0.0629	0.0669	946.24
0.3	40	0.0650	0.0688	1220.17
0.3	50	0.0658	0.0714	1571.92
0.3	60	0.0657	0.0721	1796.25
0.3	80	0.0662	0.0739	2388.12
0.3	100	0.0674	0.0756	2952.25
0.4	5	0.0718	0.0807	155.89
0.4	10	0.0665	0.0753	304.15
0.4	20	0.0631	0.0718	606.30
0.4	30	0.0689	0.0702	946.05
0.4	40	0.0736	0.0718	1219.94
0.4	50	0.0737	0.0729	1572.04
0.4	60	0.0661	0.0731	1795.88
0.4	80	0.0664	0.0735	2389.36
0.4	100	0.0682	0.0749	2950.89
0.5	5	0.0696	0.0796	156.47
0.5	10	0.0647	0.0721	304.48
0.5	20	0.0643	0.0707	605.84
0.5	40	0.0651	0.0709	1220.01
0.5	60	0.0662	0.0715	1796.10

0.5	80	0.0669	0.0719	2388.17
0.5	100	0.0685	0.0743	2951.97
0.7	5	0.0683	0.0783	156.31
0.7	10	0.0650	0.0730	303.93
0.7	20	0.0643	0.0699	605.74
0.7	40	0.0652	0.0704	1220.24
0.7	60	0.0665	0.0711	1796.31
0.7	80	0.0676	0.0723	2388.22
0.7	100	0.0694	0.0741	2952.52
0.9	5	0.0678	0.0777	156.36
0.9	10	0.0651	0.0738	304.69
0.9	20	0.0640	0.0690	605.81
0.9	40	0.0640	0.0694	1220.05
0.9	60	0.0654	0.0703	1796.41
0.9	80	0.0667	0.0710	2388.48
0.9	100	0.0722	0.0743	2952.22

TABLE C.45. RBFKOH Network RMS Error and Computation Time Behaviour with Changing Network Architecture using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application

Receptive Field Width	Hidden Layer Nodes	RMS Error		Computation Time (s)
		Train	Test	
0.1	5	0.0812	0.0910	816.52
0.1	10	0.0693	0.0789	1135.48
0.1	20	0.0665	0.0727	1864.56
0.1	40	0.0624	0.0676	3195.40
0.1	60	0.0608	0.0686	4493.80
0.1	80	0.0617	0.0713	5789.04
0.1	100	0.0645	0.0721	7109.31
0.2	5	0.0791	0.0884	816.37
0.2	10	0.0684	0.0762	1136.08
0.2	20	0.0672	0.0736	1865.29
0.2	30	0.0662	0.0694	2488.54
0.2	40	0.0634	0.0676	3195.26
0.2	50	0.0637	0.0678	3906.47
0.2	60	0.0621	0.0694	4494.49
0.2	80	0.0630	0.0719	5788.50
0.2	100	0.0648	0.0722	7110.0
0.3	5	0.0711	0.0800	816.34
0.3	10	0.0668	0.0754	1135.63
0.3	20	0.0689	0.0772	1864.42
0.3	30	0.0635	0.0671	2488.17
0.3	40	0.0613	0.0645	3195.24
0.3	50	0.0659	0.0714	3906.87
0.3	60	0.0647	0.0711	4493.83
0.3	80	0.0651	0.0724	5788.84
0.3	100	0.0664	0.0726	7108.86
0.4	5	0.0708	0.0780	815.97
0.4	10	0.0659	0.0751	1136.02
0.4	20	0.0662	0.0718	1865.27
0.4	30	0.0641	0.0671	2488.16
0.4	40	0.0715	0.0752	3195.34
0.4	50	0.0721	0.0757	3906.54
0.4	60	0.0669	0.0729	4494.15
0.4	80	0.0673	0.0738	5788.39
0.4	100	0.0681	0.0757	7109.30
0.5	5	0.0693	0.0778	816.77
0.5	10	0.0648	0.0737	1135.35

0.5	20	0.0638	0.0724	1864.36
0.5	40	0.0670	0.0764	3195.05
0.5	60	0.0670	0.0766	4493.79
0.5	80	0.0682	0.0773	5789.19
0.5	100	0.0685	0.0792	7108.04
0.7	5	0.0685	0.0772	816.63
0.7	10	0.0665	0.0760	1135.54
0.7	20	0.0664	0.0759	1864.66
0.7	40	0.0669	0.0762	3195.55
0.7	60	0.0669	0.0764	4493.40
0.7	80	0.0678	0.0771	5789.13
0.7	100	0.0689	0.0782	7109.55
0.9	5	0.0685	0.0771	816.32
0.9	10	0.0669	0.0763	1135.67
0.9	20	0.0665	0.0763	1864.40
0.9	40	0.0668	0.0769	3195.52
0.9	60	0.0676	0.0774	4493.58
0.9	80	0.0684	0.0785	5789.14
0.9	100	0.0691	0.0804	7109.02

TABLE C.46. GRNN RMS Error and Computation Time Behaviour with Changing Network Architecture using an Exponential and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Temperature Prediction Application

Receptive Field	Hidden Layer	RMS Error		Computation
Width	Nodes	Train	Test	Time (s)
0.1	10	0.1049	0.1074	0.99
0.1	50	0.0718	0.0777	3.41
0.1	100	0.0687	0.0759	6.48
0.1	200	0.0633	0.0730	12.74
0.1	400	0.0592	0.0712	25.17
0.1	500	0.0572	0.0702	31.41
0.1	600	0.0549	0.0702	37.62
0.1	700	0.0417	0.0709	43.90
0.1	800	0.0337	0.0714	49.82
0.1	1000	0.0294	0.0763	61.63
0.1	1244	0.0280	0.0815	76.95
0.2	10	0.0992	0.1045	1.02
0.2	50	0.0714	0.0795	3.42
0.2	100	0.0707	0.0796	6.49
0.2	200	0.0682	0.0779	12.71
0.2	400	0.0680	0.0785	25.19
0.2	600	0.0676	0.0789	37.59
0.2	800	0.0662	0.0791	49.79
0.2	1000	0.0655	0.0802	61.62
0.2	1244	0.0641	0.0814	77.03
0.3	10	0.0949	0.1029	0.98
0.3	50	0.0732	0.0820	3.41
0.3	100	0.0740	0.0827	6.47
0.3	200	0.0728	0.0821	12.69
0.3	400	0.0733	0.0835	25.21
0.3	600	0.0735	0.0840	37.63
0.3	800	0.0729	0.0843	49.68
0.3	1000	0.0727	0.0851	61.59
0.3	1244	0.0719	0.0859	76.94
0.5	10	0.0909	0.1006	0.99
0.5	50	0.0759	0.0855	3.43
0.5	100	0.0769	0.0866	6.46
0.5	200	0.0763	0.0866	12.73

0.5	400	0.0766	0.0874	25.23
0.5	600	0.0768	0.0876	37.63
0.5	800	0.0767	0.0881	49.78
0.5	1000	0.0766	0.0889	61.64
0.5	1244	0.0771	0.0897	76.97
0.7	10	0.0889	0.0994	1.03
0.7	50	0.0774	0.0875	3.41
0.7	100	0.0783	0.0885	6.48
0.7	200	0.0779	0.0884	12.70
0.7	400	0.0781	0.0890	25.23
0.7	600	0.0782	0.0891	37.62
0.7	800	0.0783	0.0899	49.75
0.7	1000	0.0782	0.0904	61.58
0.7	1244	0.0779	0.0916	77.06
0.9	10	0.0879	0.0988	0.98
0.9	50	0.0784	0.0886	3.39
0.9	100	0.0792	0.0895	6.51
0.9	200	0.0788	0.0894	12.73
0.9	400	0.0789	0.0899	25.18
0.9	600	0.0791	0.0900	37.59
0.9	800	0.0787	0.0908	49.76
0.9	1000	0.0792	0.0916	61.57
0.9	1244	0.0794	0.0924	76.98

TABLE C.47. WH Network Predictive Importance Results for Sigmoidal Activation Function in the Output Layer for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0623	0.0627	-	184.67
bath temperature (t-1)	0.0643	0.0685	11.933	167.34
bath resistivity	0.0674	0.0698	15.117	166.86
emf	0.0623	0.0626	1.664	167.04
cell age	0.0655	0.0661	5.171	166.95
bath height	0.0627	0.0662	5.928	167.28
high frequency noise	0.0645	0.0673	7.694	167.67
low frequency noise	0.0662	0.0682	5.996	167.32
cell power	0.0624	0.0662	5.596	167.38
AlF_3 addition	0.0625	0.0637	5.047	167.88
Na_2CO_3 addition	0.0654	0.0671	5.004	166.94
non-contributing variables omitted	0.0623	0.0626	2.217	166.89

TABLE C.48. BP1 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0620	0.0632	-	1016.92
bath temperature (t-1)	0.0641	0.0682	7.804	910.83
bath resistivity	0.0667	0.0708	10.638	909.97
emf	0.0620	0.0624	2.673	910.32
cell age	0.0619	0.0632	0.800	910.46
bath height	0.0620	0.0628	2.448	911.17
high frequency noise	0.0624	0.0655	4.181	910.62
low frequency noise	0.0621	0.0640	3.176	911.08
cell power	0.0625	0.0636	3.475	910.61
AlF_3 addition	0.0618	0.0629	0.103	909.74

Na_2CO_3 addition	0.0616	0.0623	3.249	910.55
non-contributing variables omitted	0.0617	0.0625	3.411	637.18

TABLE C.49. BP2 Network Predictive Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0553	0.0573	-	2480.24
bath temperature (t-1)	0.0591	0.0646	7.888	2263.16
bath resistivity	0.0630	0.0695	11.230	2263.54
emf	0.0616	0.0600	3.563	2262.87
cell age	0.0578	0.0604	3.794	2264.09
bath height	0.0596	0.0623	5.313	2263.29
high frequency noise	0.0588	0.0649	7.069	2263.51
low frequency noise	0.0579	0.0623	5.569	2262.84
cell power	0.0579	0.0617	4.678	2263.17
AlF_3 addition	0.0567	0.0608	3.712	2263.58
Na_2CO_3 addition	0.0562	0.0601	3.173	2262.82

TABLE C.50. RBF Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0629	0.0669	-	946.24
bath temperature (t-1)	0.0647	0.0723	8.640	934.38
bath resistivity	0.0656	0.0724	8.838	933.08
emf	0.0644	0.0665	1.690	934.23
cell age	0.0617	0.0672	2.945	934.61
bath height	0.0620	0.0673	2.681	935.09
high frequency noise	0.0631	0.0690	3.406	934.39
low frequency noise	0.0621	0.0668	0.646	933.15
cell power	0.0624	0.0666	1.630	933.92
AlF_3 addition	0.0619	0.0669	0.156	934.22
Na_2CO_3 addition	0.0617	0.0668	0.542	934.47
non-contributing variables omitted	0.0616	0.0665	1.611	897.72

TABLE C.51. RBFKOH Network Predictive Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, $\alpha = 0.01$, $df = 199$)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0613	0.0645	-	3195.24
bath temperature (t-1)	0.0658	0.0711	9.756	3173.75
bath resistivity	0.0651	0.0699	7.474	3174.68
emf	0.0644	0.0682	5.299	3173.98
cell age	0.0614	0.0644	0.970	3174.82
bath height	0.0622	0.0669	4.948	3175.11
high frequency noise	0.0652	0.0699	7.849	3174.57
low frequency noise	0.0612	0.0643	2.044	3173.86
cell power	0.0624	0.0663	3.769	3173.95
AlF_3 addition	0.0613	0.0642	2.251	3175.02
Na_2CO_3 addition	0.0611	0.0640	2.925	3174.69
non-contributing variables omitted	0.0611	0.0640	2.990	3117.81

TABLE C.52. GRNN Predictive Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Omitted from Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
no variables omitted	0.0549	0.0702	-	37.62
bath temperature (t-1)	0.0582	0.0736	7.390	36.70
bath resistivity	0.0604	0.0758	9.507	37.23
emf	0.0591	0.0707	2.683	36.81
cell age	0.0597	0.0712	4.490	36.41
bath height	0.0584	0.0715	4.767	37.19
high frequency noise	0.0565	0.0720	5.965	36.24
low frequency noise	0.0564	0.0707	3.204	37.20
cell power	0.0579	0.0718	4.774	37.18
AlF_3 addition	0.0549	0.0702	0.593	36.33
Na_2CO_3 addition	0.0549	0.0702	0.279	37.21
non-contributing variables omitted	0.0549	0.0702	0.273	34.87

TABLE C.53. WH Network Casual Importance Results for Sigmoidal Activation Function in the Output Layer for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
bath temperature (t-1)	0.0659	0.0679	8.960	167.21
bath resistivity	0.0692	0.0697	10.383	167.52
cell age	0.0636	0.0653	3.972	167.08
bath height	0.0638	0.0645	3.930	167.81
high frequency noise	0.0646	0.0665	4.784	167.09
low frequency noise	0.0643	0.0662	5.744	167.16
cell power	0.0634	0.0653	5.856	168.11
AlF_3 addition	0.0636	0.0649	5.845	167.50
Na_2CO_3 addition	0.0635	0.0656	5.961	167.64

TABLE C.54. BP1 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
bath temperature (t-1)	0.0666	0.0668	7.933	608.37
bath resistivity	0.0683	0.0685	11.900	609.16
high frequency noise	0.0642	0.0643	4.352	608.25
low frequency noise	0.0642	0.0627	2.612	608.54
cell power	0.0638	0.0629	3.247	608.49

TABLE C.55. BP2 Network Casual Importance Results using a Sigmoidal Activation Function in the Hidden and Output Layers for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in Training and Test Data Sets	RMS Error		t-statistic	Computation Time (s)
	Train	Test		
bath temperature (t-1)	0.0592	0.0642	7.066	2263.33
bath resistivity	0.0628	0.0699	12.459	2262.94
emf	0.0619	0.0599	4.058	2263.51
cell age	0.0581	0.0602	3.074	2262.18
bath height	0.0601	0.0620	5.808	2263.01
high frequency noise	0.0589	0.0647	7.909	2262.83
low frequency noise	0.0577	0.0619	6.788	2263.89
cell power	0.0578	0.0615	4.460	2262.15

AlF_3 addition	0.0570	0.0604	4.393	2262.04
Na_2CO_3 addition	0.0561	0.0598	3.014	2263.27

TABLE C.56. RBF Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
bath temperature (t-1)	0.0654	0.0753	7.411	897.13
bath resistivity	0.0661	0.0751	7.853	896.63
cell age	0.0622	0.0671	3.289	897.27
bath height	0.0623	0.0673	2.915	897.51
high frequency noise	0.0634	0.0695	4.622	896.66

TABLE C.57. RBFKOH Network Casual Importance Results using a Gaussian and Linear Activation Function in the Hidden and Output Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
bath temperature (t-1)	0.0684	0.0725	10.395	3118.14
bath resistivity	0.0678	0.0712	7.242	3118.00
emf	0.0654	0.0693	5.021	3117.87
bath height	0.0635	0.0672	4.697	3117.90
high frequency noise	0.0681	0.0714	7.294	3117.91
cell power	0.0629	0.0667	4.445	3118.14

TABLE C.58. GRNN Casual Importance Results using a Gaussian and Linear Activation Function in the Pattern and Summation Layers, Respectively, for Electrolyte Temperature Prediction Application (t-critical = 2.345, α = 0.01, df = 199)

Input Variable Varied in	RMS Error			Computation
Training and Test Data Sets	Train	Test	t-statistic	Time (s)
bath temperature (t-1)	0.0583	0.0781	9.552	34.85
bath resistivity	0.0597	0.0814	13.156	35.19
emf	0.0581	0.0715	3.151	34.86
cell age	0.0589	0.0729	5.033	35.05
bath height	0.0570	0.0743	6.069	34.81
high frequency noise	0.0547	0.0760	8.433	34.90
low frequency noise	0.0552	0.0711	3.163	34.88
cell power	0.0561	0.0736	5.013	35.11

TABLE C.59. Percentage Contribution of Input Variables using Predictive Importance Results for Electrolyte Temperature Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
bath temperature (t-1)	14.9	31.1	13.6	39.4	26.1	21.7
bath resistivity	18.3	47.2	22.8	40.1	21.3	35.7
emf	0.0	0.0	5.0	0.0	14.6	3.2
cell age	8.8	0.0	5.8	2.2	0.0	6.4
bath height	9.0	0.0	9.3	2.9	9.5	8.3
high frequency noise	11.9	14.3	14.2	15.3	21.3	11.5
low frequency noise	14.2	3.5	9.3	0.0	0.0	3.2
cell power	9.0	4.0	8.2	0.0	7.1	10.2
AlF_3 addition	2.6	0.0	6.5	0.0	0.0	0.0
Na_2CO_3 addition	11.3	0.0	5.2	0.0	0.0	0.0

TABLE C.60. Percentage Contribution of Input Variables using Casual Importance Results for Electrolyte Temperature Prediction Application

	Neural Network Model					
Parameter	WH	BP1	BP2	RBF	RBFKOH	GRNN
bath temperature (t-1)	16.3	32.1	13.4	40.4	24.8	21.2
bath resistivity	21.8	42.6	24.5	39.4	21.0	30.0
emf	-	-	5.0	-	15.5	3.5
cell age	8.3	-	5.6	2.8	-	7.2
bath height	6.7	-	9.1	3.7	9.3	11.0
high frequency noise	12.0	16.7	14.4	13.8	21.6	15.5
low frequency noise	11.1	3.8	8.9	-	-	2.4
cell power	8.3	4.9	8.2	-	7.9	9.1
AlF_3 addition	5.2	-	6.0	-	-	-
Na_2CO_3 addition	10.2	-	4.9	-	-	-

APPENDIX D

Neural Network Analysis and Optimisation Strategy User's Guide

D.1 NEURAL NETWORK ANALYSIS AND OPTIMISATION STRATEGY USER'S GUIDE

This user's guide is written to provide an overview of the capabilities of the *Neural Network Analysis and Optimisation Strategy* program accompanying this documentation and moreover, give the user the best opportunity to use the program efficiently and productively. The *Neural Network Analysis and Optimisation Strategy* program is designed to complete all the necessary data manipulation required for quantitative and comprehensive neural network analysis. In particular, the analysis package can be used to quickly and efficiently calculate the minimum and maximum values of each parameter in a specified number of data patterns, normalise, or scale, the data patterns using a specified minimum-maximum range and divide the data patterns into train and test data sets. Further, the program is designed to facilitate an analysis of input parameter importance, including the option to rank the input parameters from highest to lowest significance with the percentage contribution of each input calculated and documented on the user interface. Moreover, the program incorporates procedures to determine the optimum combination of prediction error, number of input parameters and computation time to maximise the economic benefit of neural network modelling. The optimisation part of the *Neural Network Analysis and Optimisation Strategy* program has been exclusively developed to determine an optimal solution for a neural network model using a specific technique not offered by commercially available packages.

The user is prompted to input particular data as required depending on the analysis being completed. However, it is useful to note that the program is designed to minimise the amount of data that is required to be entered by the user. In particular,

for the majority of neural network model analysis the user is required only to enter the parameter names and the data patterns associated with the application. If an optimal solution is required for a neural network model then the cost associated with the decision variables representing error, input parameters and computation time must be entered. However, the program has been carefully written to minimise the amount of manual data entry required by the user.

The *Neural Network Analysis and Optimisation Strategy* program is named *NetAnal.xls* on the accompanying disk and is executable using Microsoft® Excel, version 97 SR-2, or later. The program is stored in the directory *Opt_Prog* in the sub-directory *Program*. Figure D.1.1 shows a typical Microsoft® Excel spreadsheet with some important features labelled to familiarise the user with some common spreadsheet terminology. It is necessary to note these features here as they will be referred to throughout this user guide.

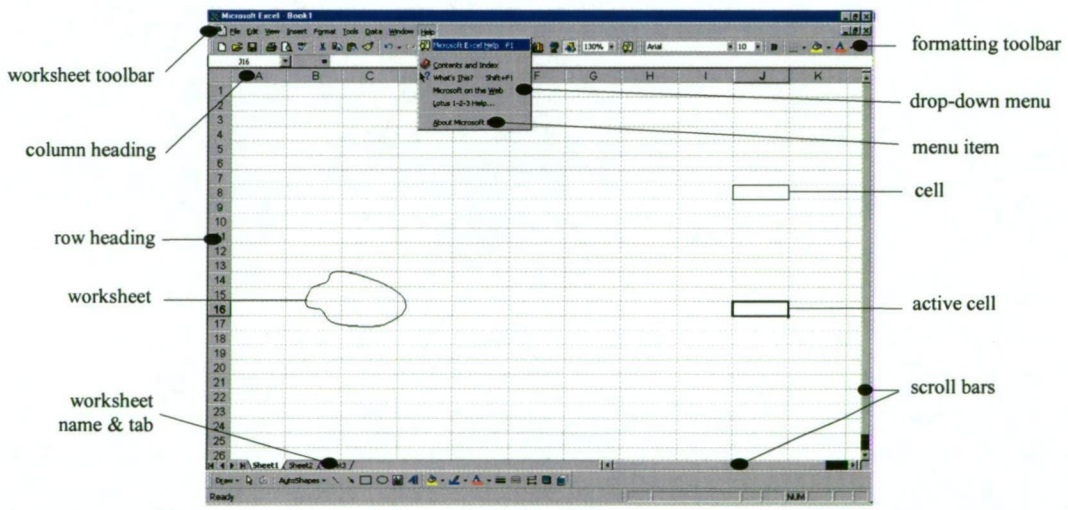


Fig. D.1.1. Typical Microsoft® Excel Spreadsheet Layout Highlighting Some Important Features

The *Neural Network Analysis and Optimisation Strategy* program is carefully written to be user-friendly and interacts with the user through the Windows graphical user interface (GUI). Further, menus and toolbars are utilised as they provide a quick, convenient and widely accessible way to expose commands and options to the user.

D.1.1 Starting the Neural Network Analysis and Optimisation Strategy Program

Select *Open* from the *File* menu on the worksheet toolbar in Microsoft® Excel and map to the appropriate directory where *NetAnal.xls* is stored. Select the file *NetAnal.xls* and click *Open*, or alternatively, double-click on the file name to open it. Alternatively, from the computer desktop, map to the directory containing the file and select *NetAnal.xls* to open it without first opening Microsoft® Excel. Select *Enable Macros* on the message box that appears on the user interface. As a result, the *Neural Network Analysis and Optimisation Strategy* program is enabled. In the first instance, an introductory screen is presented on the user interface, as shown in Figure D.1.2. Click on *START NEURAL NETWORK ANALYSIS AND OPTIMISATION STRATEGY* to proceed.

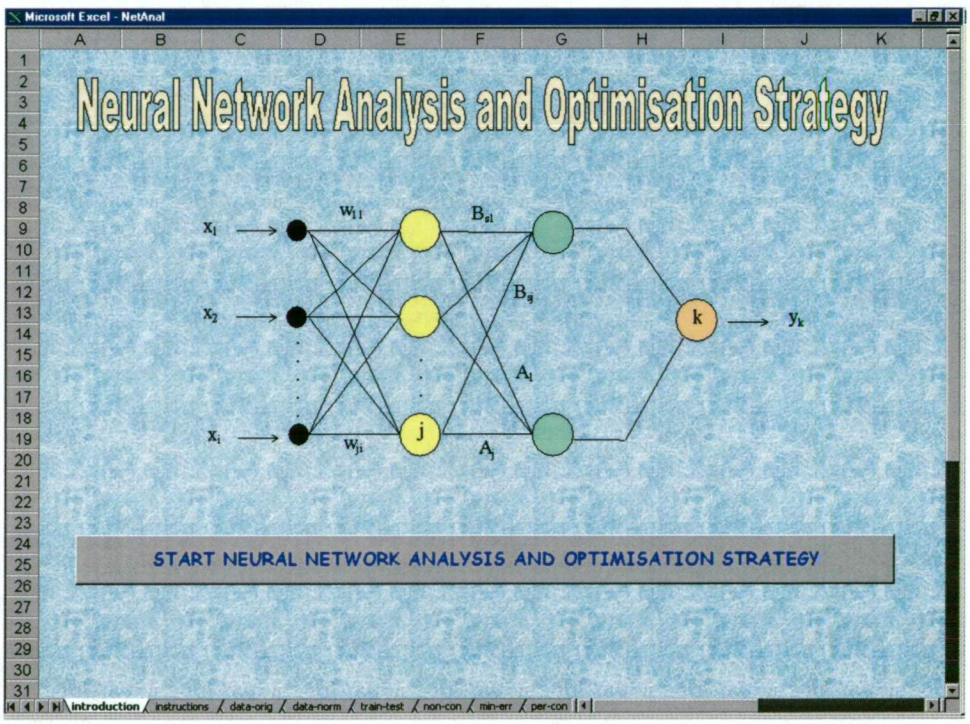


Fig. D.1.2. *Neural Network Analysis and Optimisation Strategy* Introductory Screen Displayed on User Interface

D.1.2 An Important Note to the Neural Network Analysis and Optimisation Strategy Program User

The user screen shown in Figure D.1.3 is displayed by clicking on *START NEURAL NETWORK ANALYSIS AND OPTIMISATION STRATEGY*. The brief instructions provide an overview of the program, user data entry requirements and a copyright

notice. Click on *I have read this important note and am ready to advance* to proceed with the program.

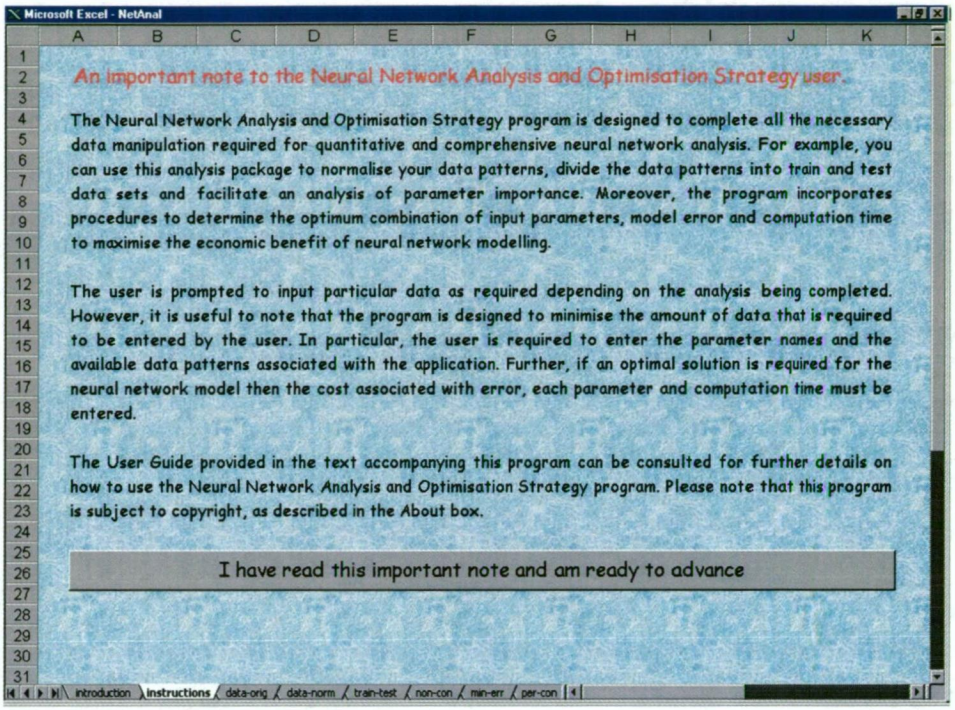


Fig. D.1.3. *Neural Network Analysis and Optimisation Strategy* Instruction Screen Displayed on User Interface

As a result of this command the *data-orig* worksheet is activated and a customised toolbar, *NeuralNet*, is positioned at the top of the worksheet and enabled. The objective of enabling the *NeuralNet* toolbar only after the user has selected *I have read this important note and am ready to advance* is to prompt the user to read the important note before proceeding. The *NeuralNet* toolbar contains a series of drop-down menus, each of which contain a series of menu items necessary for neural network model analysis and optimisation. All other built-in toolbars are hidden in order to maximise the available space on the user interface. However, any built-in toolbar can be enabled at any time by selecting, from the *NeuralNet* toolbar, *Microsoft Excel*, *View*, *Toolbars* and clicking on the desired choice.

In addition to activating the *data-orig* worksheet and enabling the *NeuralNet* toolbar, a message box is presented on the user interface detailing instructions for proceeding with the program, as shown in Figure D.1.4.

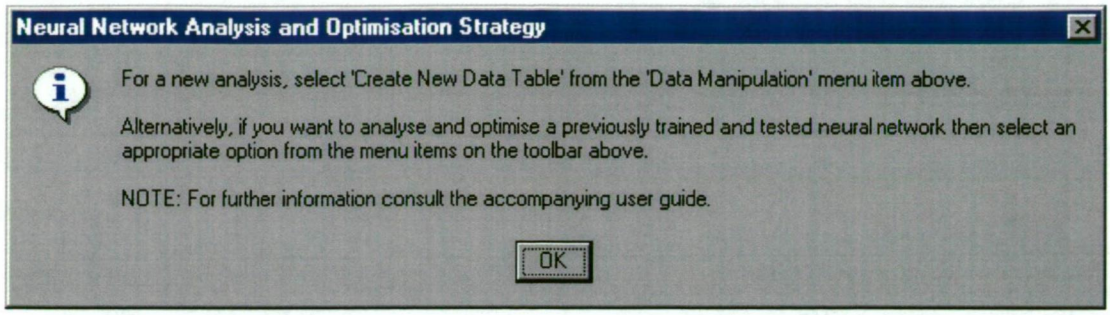


Fig. D.1.4. Message Box Displayed on User Interface Detailing Instructions for Proceeding with the Program

D.1.3 Selecting a Menu Item from the *NeuralNet* Toolbar

Select a menu item from the *NeuralNet* toolbar by clicking the mouse button as appropriate or alternatively, use the *Alt + (underlined character)* keys on the keyboard to select the desired menu item. Each drop-down menu on the *NeuralNet* toolbar has a series of menu items that can be selected to perform a desired task. The *Text Files* drop-down menu item and associated menu items are shown in Figure D.1.5 as an illustration of the *NeuralNet* toolbar. The function of each drop-down menu on the *NeuralNet* toolbar and the associated menu items are discussed in the following section.

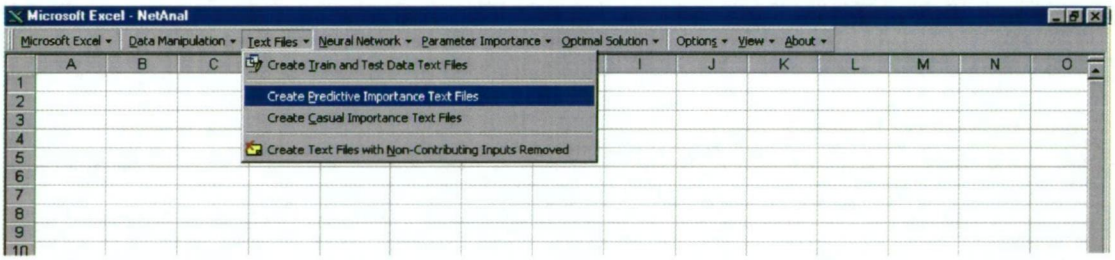
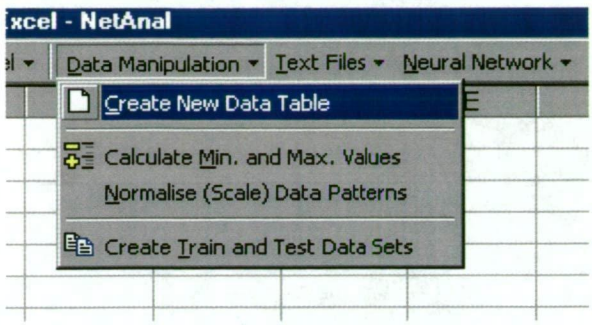


Fig. D.1.5. Illustration of *NeuralNet* Toolbar and Menu Item Selection Displayed on User Interface

Microsoft Excel▾

All menu items associated with this particular drop-down menu are standard Microsoft® Excel features and have been included to allow manipulation of the active worksheet and execution of specific commands as appropriate. It is assumed that the user is familiar with Microsoft® Excel, hence, the standard features included in the *Neural Network Analysis and Optimisation Strategy* are not discussed here.

Data Manipulation ▾



The *Data Manipulation* menu is used specifically for entering parameter names and data patterns associated with a particular application into the workbook and subsequent data manipulation. Data manipulation includes establishment of the minimum and maximum values of each parameter, normalising the data patterns and creating appropriate train and test data sets.

Create New Data Table - For a new application this is the starting point for neural network model analysis and optimisation. It is where the user creates a suitable table for entering the parameter names and data patterns associated with the application. Selecting *Create New Data Table* from the menu items displays a message box to the user interface requesting the number of input and output variables and data patterns associated with the application, as shown in Figure D.1.6.

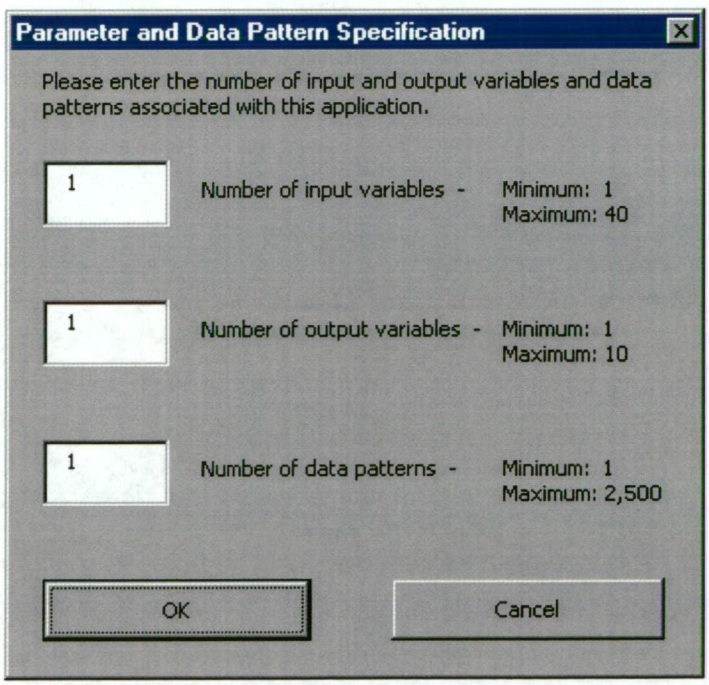
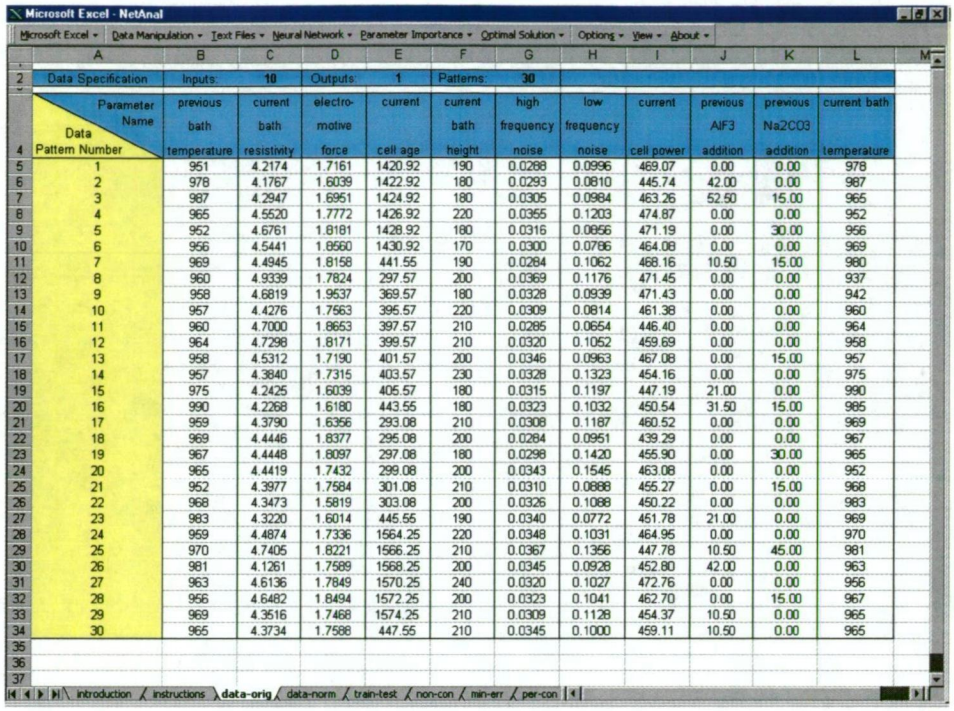


Fig. D.1.6. Parameter and Data Pattern Specification Message Box Requesting Information to Create New Data Table

Enter the number of input and output variables and data patterns and select *OK* to create a data table ready for data entry. While the minimum number of input and output parameters and data patterns are limited to 1, 1 and 1 respectively, there is also a maximum limit on the number of input and output parameters and data patterns allowed. The maximum limits have been set to coincide with existing computer processing capability. In particular, the maximum limits specified here allow the accompanying neural network programs to be trained and tested without the occurrence of a floating point overflow error, which typically occurs as a result of information overload. Hence, the maximum number of input parameters is 40 while the maximum number of output parameters is 10. Further, the maximum number of data patterns is set at 2,500. However, it is useful to note that for typical industrial applications this limitation is of no disadvantage as the associated number of input and output parameters and data patterns are typically significantly lower than the specified limitations. Moreover, a typical industrial application generally requires no more than 15-20 input parameters, 1-5 output parameters and no more than 2,000-2,500 data patterns.

Once the number of input and output parameters and data patterns have been specified an appropriate data table is created on the active worksheet. Enter the parameter names horizontally across the worksheet and the data patterns vertically down the worksheet. For example, for 10 input parameters, 1 output parameter and 30 data patterns, data is entered as shown in Figure D.1.7. The number of input and output parameters and data patterns are displayed at the top of the table for reference. A function of this menu item is a prompt to open a workbook to copy any relevant data from.

Calculate Min. and Max. Values - This particular menu item is used to calculate the minimum and maximum values of each parameter associated with the application. The minimum and maximum values are displayed on the user screen, as shown in Figure D.1.8(a). However, the minimum and maximum values cannot be calculated if the data table has not been created and the relevant data entered. Further, prior to calculating the minimum and maximum values the data patterns are searched for missing data. If a blank cell is identified, the user is prompted to enter a value before the minimum and maximum values can be calculated, as shown in Figure D.1.8(b).



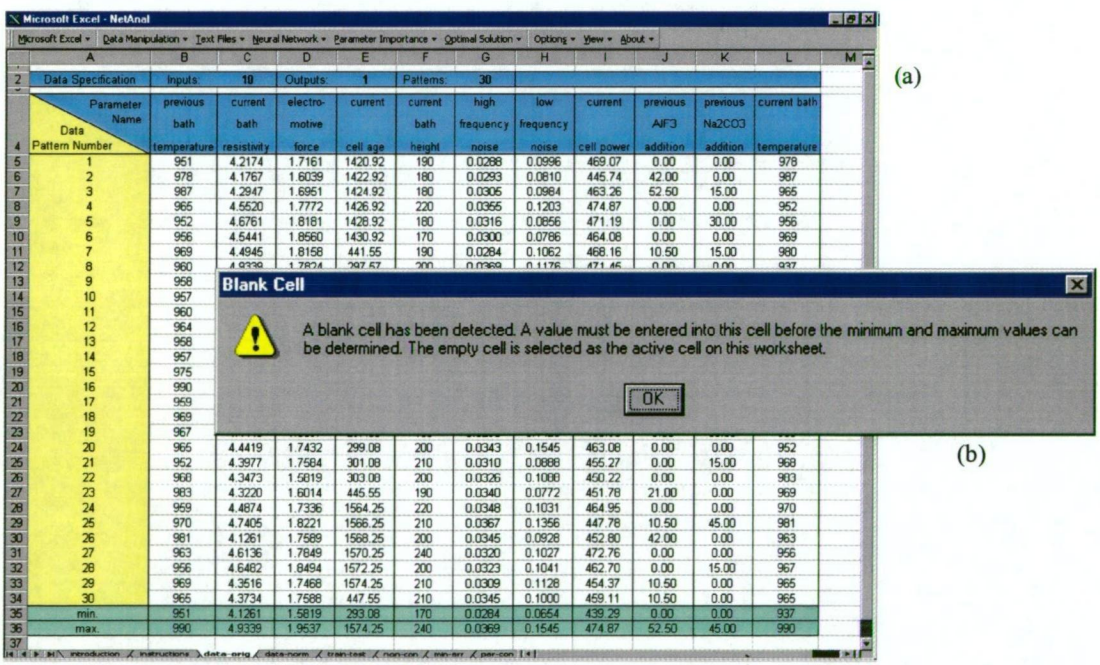
Microsoft Excel - NetAnal

Microsoft Excel | Data Manipulation | Text Files | Neural Network | Parameter Importance | Optimal Solution | Options | View | About

	A	B	C	D	E	F	G	H	I	J	K	L
2	Data Specification	Inputs:	10	Outputs:	1	Patterns:	30					
4	Parameter Name / Data Pattern Number	previous bath temperature	current bath resistivity	electro-motive force	current cell age	current bath height	high frequency noise	low frequency noise	current cell power	previous AlF3 addition	previous Na2CO3 addition	current bath temperature
5	1	951	4.2174	1.7161	1420.92	190	0.0288	0.0996	469.07	0.00	0.00	978
6	2	978	4.1767	1.6039	1422.92	180	0.0293	0.0810	445.74	42.00	0.00	987
7	3	987	4.2947	1.6951	1424.92	180	0.0305	0.0984	463.26	52.50	15.00	965
8	4	965	4.5520	1.7772	1426.92	220	0.0355	0.1203	474.87	0.00	0.00	952
9	5	952	4.6761	1.8181	1428.92	180	0.0316	0.0856	471.19	0.00	30.00	956
10	6	956	4.5441	1.8560	1430.92	170	0.0300	0.0786	464.08	0.00	0.00	969
11	7	969	4.4945	1.8158	441.55	190	0.0284	0.1062	468.16	10.50	15.00	980
12	8	960	4.9339	1.7824	297.57	200	0.0369	0.1176	471.45	0.00	0.00	937
13	9	958	4.6819	1.9537	369.57	180	0.0328	0.0939	471.43	0.00	0.00	942
14	10	957	4.4276	1.7563	395.57	220	0.0309	0.0814	461.38	0.00	0.00	960
15	11	960	4.7000	1.8653	397.57	210	0.0285	0.0654	446.40	0.00	0.00	964
16	12	964	4.7298	1.8171	399.57	210	0.0320	0.1052	459.69	0.00	0.00	958
17	13	958	4.5312	1.7190	401.57	200	0.0346	0.0963	467.08	0.00	15.00	957
18	14	957	4.3840	1.7315	403.57	230	0.0328	0.1323	454.16	0.00	0.00	975
19	15	975	4.2425	1.6039	405.57	180	0.0315	0.1197	447.19	21.00	0.00	990
20	16	990	4.2268	1.6180	443.55	180	0.0323	0.1032	450.54	31.50	15.00	985
21	17	959	4.3790	1.6356	293.08	210	0.0308	0.1187	460.52	0.00	0.00	969
22	18	969	4.4446	1.8377	295.08	200	0.0284	0.0951	439.29	0.00	0.00	967
23	19	967	4.4448	1.8097	297.08	180	0.0298	0.1420	455.90	0.00	30.00	965
24	20	965	4.4419	1.7432	299.08	200	0.0343	0.1545	463.08	0.00	0.00	952
25	21	952	4.3977	1.7584	301.08	210	0.0310	0.0888	455.27	0.00	15.00	968
26	22	968	4.3473	1.5819	303.08	200	0.0326	0.1088	450.22	0.00	0.00	983
27	23	983	4.3220	1.6014	445.55	190	0.0340	0.0772	451.78	21.00	0.00	969
28	24	959	4.4874	1.7336	1564.25	220	0.0348	0.1031	464.95	0.00	0.00	970
29	25	970	4.7405	1.8221	1566.25	210	0.0367	0.1356	447.78	10.50	45.00	981
30	26	981	4.1261	1.7589	1568.25	200	0.0345	0.0928	452.80	42.00	0.00	963
31	27	963	4.6136	1.7849	1570.25	240	0.0320	0.1027	472.76	0.00	0.00	956
32	28	956	4.6482	1.8494	1572.25	200	0.0323	0.1041	462.70	0.00	15.00	967
33	29	969	4.3516	1.7468	1574.25	210	0.0309	0.1128	454.37	10.50	0.00	965
34	30	965	4.3734	1.7588	447.55	210	0.0345	0.1000	459.11	10.50	0.00	965
35												
36												
37												

introduction | instructions | data-orig | data-norm | train-test | non-con | min-err | per-con

Fig. D.1.7. Data Table Showing Parameter Names and Data Patterns (10 Input Parameters, 1 Output Parameter and 30 Data Patterns)



(a)

Microsoft Excel - NetAnal

Microsoft Excel | Data Manipulation | Text Files | Neural Network | Parameter Importance | Optimal Solution | Options | View | About

	A	B	C	D	E	F	G	H	I	J	K	L
2	Data Specification	Inputs:	10	Outputs:	1	Patterns:	30					
4	Parameter Name / Data Pattern Number	previous bath temperature	current bath resistivity	electro-motive force	current cell age	current bath height	high frequency noise	low frequency noise	current cell power	previous AlF3 addition	previous Na2CO3 addition	current bath temperature
5	1	951	4.2174	1.7161	1420.92	190	0.0288	0.0996	469.07	0.00	0.00	978
6	2	978	4.1767	1.6039	1422.92	180	0.0293	0.0810	445.74	42.00	0.00	987
7	3	987	4.2947	1.6951	1424.92	180	0.0305	0.0984	463.26	52.50	15.00	965
8	4	965	4.5520	1.7772	1426.92	220	0.0355	0.1203	474.87	0.00	0.00	952
9	5	952	4.6761	1.8181	1428.92	180	0.0316	0.0856	471.19	0.00	30.00	956
10	6	956	4.5441	1.8560	1430.92	170	0.0300	0.0786	464.08	0.00	0.00	969
11	7	969	4.4945	1.8158	441.55	190	0.0284	0.1062	468.16	10.50	15.00	980
12	8	960	4.9339	1.7824	297.57	200	0.0369	0.1176	471.45	0.00	0.00	937
13	9	958										
14	10	957										
15	11	960										
16	12	964										
17	13	958										
18	14	957										
19	15	975										
20	16	990										
21	17	959										
22	18	969										
23	19	967										
24	20	965	4.4419	1.7432	299.08	200	0.0343	0.1545	463.08	0.00	0.00	952
25	21	952	4.3977	1.7584	301.08	210	0.0310	0.0888	455.27	0.00	15.00	968
26	22	968	4.3473	1.5819	303.08	200	0.0326	0.1088	450.22	0.00	0.00	983
27	23	983	4.3220	1.6014	445.55	190	0.0340	0.0772	451.78	21.00	0.00	969
28	24	959	4.4874	1.7336	1564.25	220	0.0348	0.1031	464.95	0.00	0.00	970
29	25	970	4.7405	1.8221	1566.25	210	0.0367	0.1356	447.78	10.50	45.00	981
30	26	981	4.1261	1.7589	1568.25	200	0.0345	0.0928	452.80	42.00	0.00	963
31	27	963	4.6136	1.7849	1570.25	240	0.0320	0.1027	472.76	0.00	0.00	956
32	28	956	4.6482	1.8494	1572.25	200	0.0323	0.1041	462.70	0.00	15.00	967
33	29	969	4.3516	1.7468	1574.25	210	0.0309	0.1128	454.37	10.50	0.00	965
34	30	965	4.3734	1.7588	447.55	210	0.0345	0.1000	459.11	10.50	0.00	965
35	min.	951	4.1261	1.5819	293.08	170	0.0284	0.0654	439.29	0.00	0.00	937
36	max.	990	4.9339	1.9537	1574.25	240	0.0369	0.1545	474.87	52.50	45.00	990
37												

introduction | instructions | data-orig | data-norm | train-test | non-con | min-err | per-con

Blank Cell

A blank cell has been detected. A value must be entered into this cell before the minimum and maximum values can be determined. The empty cell is selected as the active cell on this worksheet.

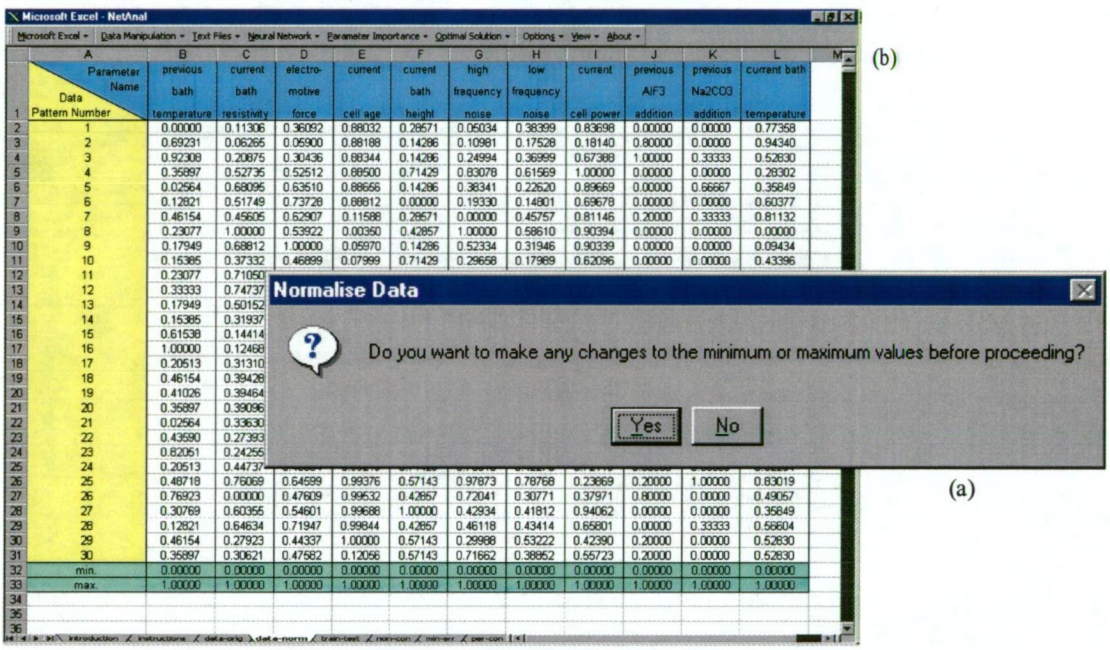
OK

(b)

Fig. D.1.8. (a) Minimum and Maximum Values of Data Patterns, and (b) Message Box Notification that Required Data is Missing

Normalise (Scale) Data Patterns - It is important to note that the minimum and maximum values are not automatically calculated, if not calculated prior, during this

procedure. It is often necessary to extend the range of the minimum and maximum values in order to allow for slightly higher and lower values of the associated parameters that may be encountered when the neural network model is implemented. Hence, the user is prompted to alter the minimum and maximum values as desired, as shown in Figure D.1.9(a). Therefore, calculation of the minimum and maximum values is a separate procedure. The normalised data patterns are displayed on the user interface, as shown in Figure D.1.9(b). The minimum and maximum values of each parameter in the data patterns are displayed on the normalised data pattern table. This is a useful indication of the suitability of the range of the specified minimum and maximum values.



Parameter Name / Data Pattern Number	previous bath temperature	current bath resistivity	electro-motive force	current cell age	current bath height	high frequency noise	low frequency noise	current cell power	previous AlF3 addition	previous Na2CO3 addition	current bath temperature
1	0.00000	0.11306	0.36092	0.88032	0.28571	0.05034	0.38399	0.83698	0.00000	0.00000	0.77358
2	0.69231	0.06265	0.05900	0.88188	0.14286	0.10981	0.17528	0.18140	0.80000	0.00000	0.94340
3	0.92308	0.20875	0.30436	0.88344	0.14286	0.24994	0.36999	0.67388	1.00000	0.33333	0.52830
4	0.35897	0.52735	0.52512	0.88500	0.71429	0.83078	0.61569	1.00000	0.00000	0.00000	0.28302
5	0.02564	0.68095	0.63510	0.88656	0.14286	0.38341	0.22620	0.89669	0.00000	0.66667	0.35849
6	0.12821	0.51749	0.73728	0.88812	0.00000	0.19330	0.14801	0.69678	0.00000	0.00000	0.60377
7	0.46154	0.45605	0.62907	0.11588	0.28571	0.00000	0.45757	0.81146	0.20000	0.33333	0.81132
8	0.23077	1.00000	0.53922	0.00350	0.42857	1.00000	0.58610	0.90394	0.00000	0.00000	0.00000
9	0.17949	0.68812	1.00000	0.05970	0.14286	0.52334	0.31946	0.90339	0.00000	0.00000	0.09434
10	0.15385	0.37332	0.46899	0.07999	0.71429	0.29658	0.17989	0.62096	0.00000	0.00000	0.43396
11	0.23077	0.71050									
12	0.33333	0.74737									
13	0.17949	0.50152									
14	0.15385	0.31937									
15	0.61538	0.14414									
16	1.00000	0.12468									
17	0.20513	0.31310									
18	0.46154	0.39428									
19	0.41026	0.39464									
20	0.35897	0.39096									
21	0.02564	0.33630									
22	0.43590	0.27393									
23	0.82051	0.24255									
24	0.20513	0.44737									
25	0.48718	0.76069	0.64599	0.99376	0.57143	0.97873	0.78768	0.23869	0.20000	1.00000	0.83019
26	0.76923	0.00000	0.47609	0.99532	0.42857	0.72041	0.30771	0.37971	0.80000	0.00000	0.49057
27	0.30769	0.60355	0.54601	0.99688	1.00000	0.42934	0.41812	0.94062	0.00000	0.00000	0.35849
28	0.12821	0.64634	0.71947	0.99844	0.42857	0.46118	0.43414	0.65801	0.00000	0.33333	0.56604
29	0.46154	0.27923	0.44337	1.00000	0.57143	0.29988	0.53222	0.42390	0.20000	0.00000	0.52830
30	0.35897	0.30621	0.47582	0.12056	0.57143	0.71662	0.38852	0.55723	0.20000	0.00000	0.52830
min.	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
max.	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000

Fig. D.1.9. (a) Message Box Notification to Modify Minimum and Maximum Values, and (b) Normalised Data Patterns

Create Train and Test Data Sets - Providing the data patterns have been normalised and there are no blank cells among the data patterns, a procedure is used to ensure this, the data patterns can be manually or randomly divided into train and test data sets. The user is presented with the message boxes shown in Figure D.1.10 prompting a decision of manual or random selection of the train and test data sets from the specified data patterns. Subsequently, the user is required to specify the required number of test data patterns.

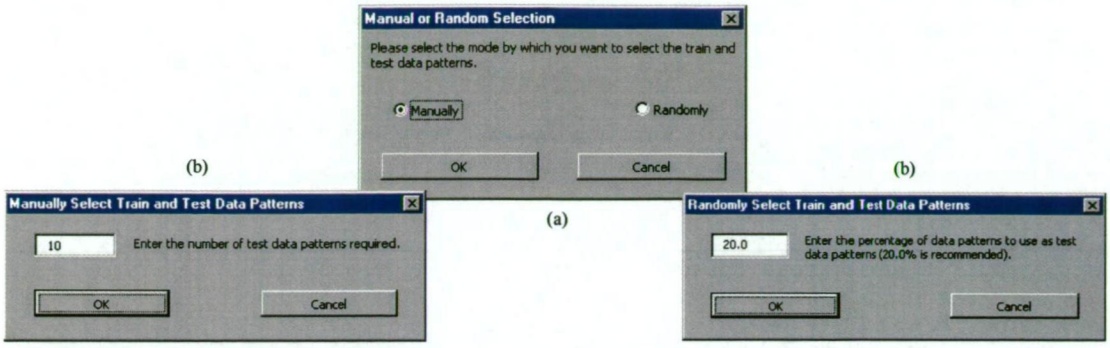


Fig. D.1.10. Message Box Displayed on User Interface Requesting User Information Regarding (a) Manual or Random Selection of Train and Test Data Sets, and (b) Number of Test Data Patterns Required

Manually - If there are specific data patterns that are to be used as train and test data sets, select *Manually* on the custom dialogue box. Consequently, an empty data table is prepared to accommodate the required number of train and test data patterns, ready for data pattern entry, as shown in Figure D.1.11(a).

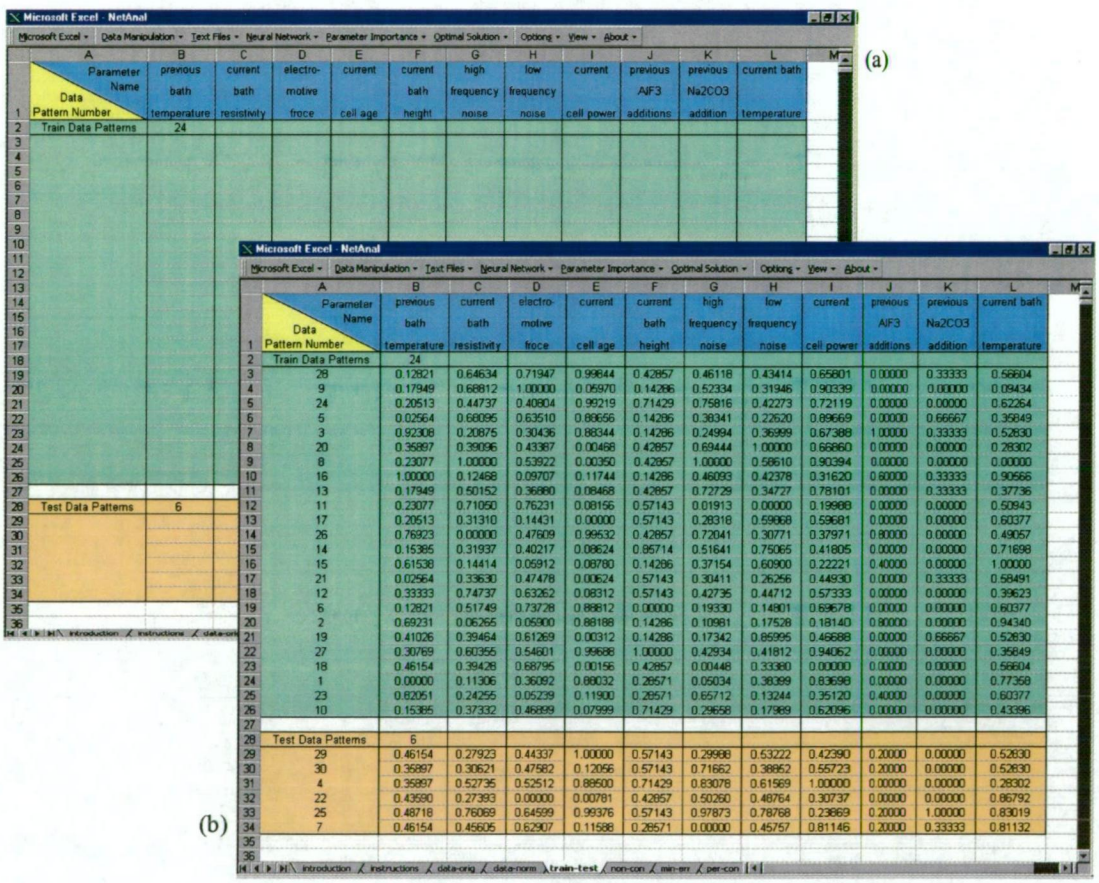
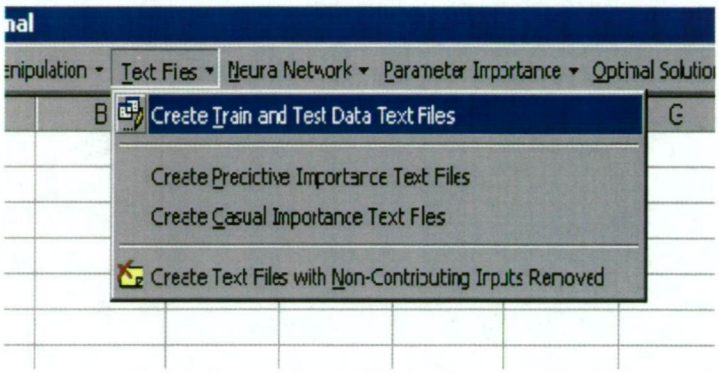


Fig. D.1.11. Train and Test Data Sets (a) Manual Pattern Entry, and (b) Randomly Selected from Specified Data Patterns

Randomly - If there is no requirement to use specific data patterns as the train and test data sets, select *Randomly* on the custom dialogue box. Consequently, the data patterns are randomly divided into the required number of train and test patterns, as shown in Figure D.1.11(b).

Text Files ▾

The menu items available in this drop-down menu are used to create train and test data text files that are compatible with the accompanying neural network programs. The text files created using the menu items available with this drop-down menu are used for neural network training and testing and completing a predictive or casual importance analysis.



Create Train and Test Data Text Files - This menu item is used to create the train and test data text files with all potential input and output parameters included in the model. While a procedure is used to examine the train and test data patterns for blank cells, if there is no missing data the user is required to specify the directory to save the train and test data text files to. Consequently, the train and test data text files are saved with the identifiers 'trn.txt' and 'tst.txt', respectively, and have the format 'Text (Tab-Delimited)', as shown in Figure D.1.12.

Create Predictive Importance Text Files - The predictive importance train and test data text files are created from the normalised train and test data patterns. There are i train data text files and i test data text files created, where i is the number of input parameters. The train and test data text files are saved using the identifiers 'pitrn1.txt' and 'pitst1.txt', 'pitrn2.txt' and 'pitst2.txt', ..., 'pitrn*n*.txt' and 'pitst*n*.txt', where n = 1, ..., i, and are saved to a directory specified by the user. The text files 'pitrn1.txt' and 'pitst1.txt' are unique as they have the first input parameter omitted from the data sets, while all other input and output parameters are maintained, whereas 'pitrn*n*.txt' and 'pitst*n*.txt' are unique as they have the *nth* input parameter omitted from the data

sets, while all other input and output parameters are maintained. Hence, the identifying number of the train and test data text files indicates the input parameter omitted from the train and test data sets. The format of the text files is the same as that shown in Figure D.1.12.

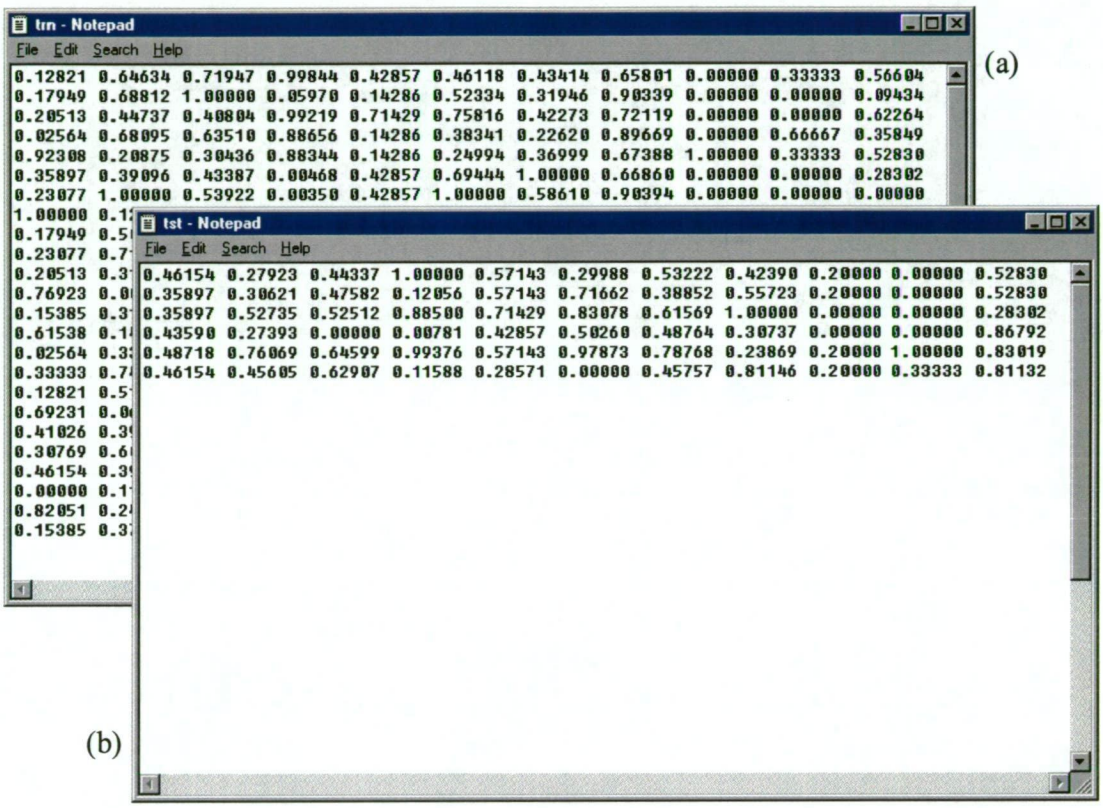


Fig. D.1.12. Data Text Files (a) 24 Train Data Patterns, and (b) 6 Test Data Patterns

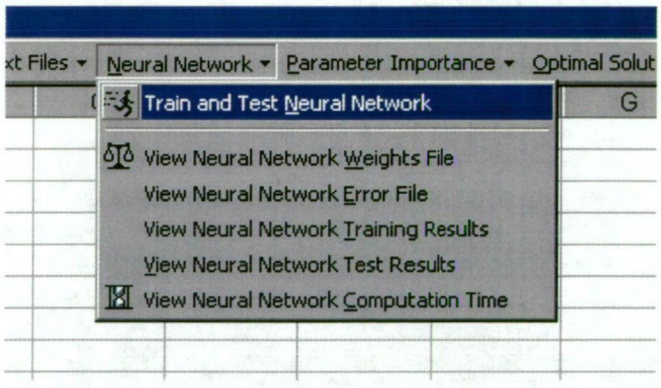
Create Casual Importance Text Files - The casual importance train and test data text files are created from the normalised train and test data patterns. There are *i* train data text files and *i* test data text files created, where *i* is the number of input parameters. The train and test data text files are saved using the identifiers 'citrn1.txt' and 'citst1.txt', 'citrn2.txt' and 'citst2.txt', …, 'citrn*n*.txt' and 'citst*n*.txt', where $n = 1, ..., i$, and are saved to a directory specified by the user. The text files 'citrn1.txt' and 'citst1.txt' are unique as they have the first input parameter varied over the bounded range 0.0 to 1.0, while all other input and output parameters are maintained, whereas 'citrn*n*.txt' and 'citst*n*.txt' are unique as they have the *nth* input parameter varied over the bounded range 0.0 to 1.0, while all other input and output parameters are maintained. Hence, the identifying number of the train and test data text files indicates the input parameter that is varied over the bounded range 0.0 to 1.0. While

each input parameter is varied over the bounded range 0.0 to 1.0, the value assigned to each parameter is a function of the number of data patterns and the data pattern number. The input parameter in the first and last data pattern is assigned the value 0.0 and 1.0, respectively, while the value of the input parameter in the second and subsequent data patterns is equivalent to the value of the previous data pattern plus an increment. For the train data set the size of the increment is equivalent to the inverse of the number of train data patterns, num_{trn}, minus one, ie. $(num_{trn} - 1)^{-1}$, while the size of the increment for the test data patterns is equivalent to the inverse of the number of test data patterns, num_{tst}, minus one, ie. $(num_{tst} - 1)^{-1}$. The format of the text files is the same as that shown in Figure D.1.12.

Create Text Files with Non-Contributing Inputs Removed - A prerequisite for this command is establishment of the minimum test error from the casual or predictive importance analysis. The results of this analysis are required to establish any non-contributing input parameters in the model. If this condition is not satisfied, the user is requested to do so before this command can proceed. The normalised train and test data patterns are copied to a new worksheet and any non-contributing input parameters are removed from the train and test data sets. Subsequently, the train and test data text files are created and saved to a directory specified by the user. The train and test data text files are saved using the identifiers 'trnnon.txt' and 'tstnon.txt'. The format of the text files is the same as that shown in Figure D.1.12.

Neural Network ▾

The menu items associated with this particular drop-down menu item are used to determine the values of the constants that require specification in the accompanying neural network programs. In addition, use the menu items in this drop-down menu to view the neural network weights, error, training and test results and computation time files.



Train and Test Neural Network - The accompanying neural network programs are specifically designed to minimise the manual data entry required by the user. However, some constants require values to be specified within the program in order to customise a neural network model for a particular application. Hence, to minimise the opportunity for an error to occur during data entry into the desired neural network program, the values of the constants requiring specification are displayed on the user interface. Prior to this the user is required to select the neural network model required for training and testing through use of a message box displayed on the user interface, as shown in Figure D.1.13.

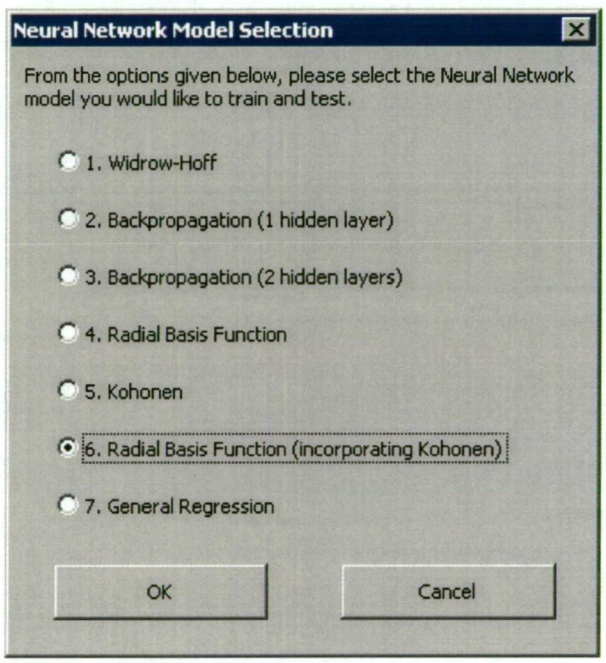


Fig. D.1.13. Message Box Displayed on User Interface Requesting Neural Network Model Selection

Subsequently, the custom dialogue box shown in Figure D.1.14 is displayed on the user interface highlighting the relevant constants that require specification in the selected neural network program and the appropriate values to enter. The values of the specified constants are dependent on the number of input and output parameters and the number of train and test data patterns. Further, the appropriate Pascal program to open for the selected neural network model is displayed on the custom dialogue box.

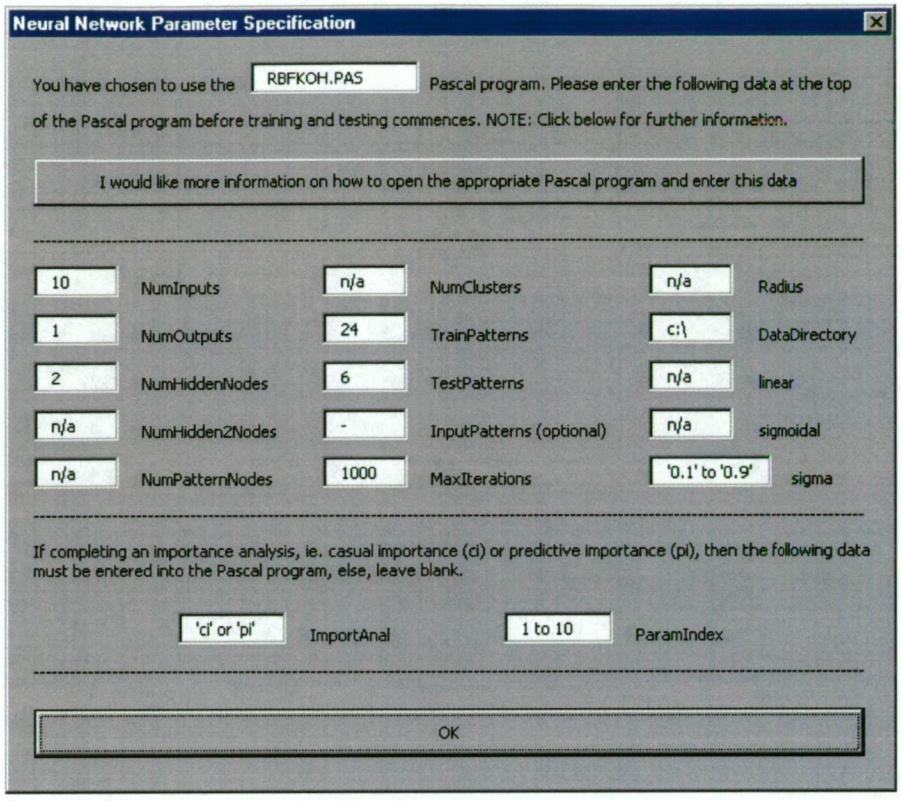


Fig. D.1.14. Custom Dialogue Box Displayed on User Interface Highlighting Neural Network Parameter Specification

If immediate 'on-line' information is required in order to open the appropriate neural network program and enter the necessary data, click on *I would like more information on how to open the appropriate Pascal program and enter this data.* The step-by-step guide shown in Figure D.1.15 is displayed on the user interface as a result of this action. However, the information provided on this custom dialogue box is only a brief guide; consult Appendix A for a detailed explanation of the procedure required to edit and run the accompanying neural network programs. However, some information is provided here for entering the values of the specified constants into the appropriate neural network program.

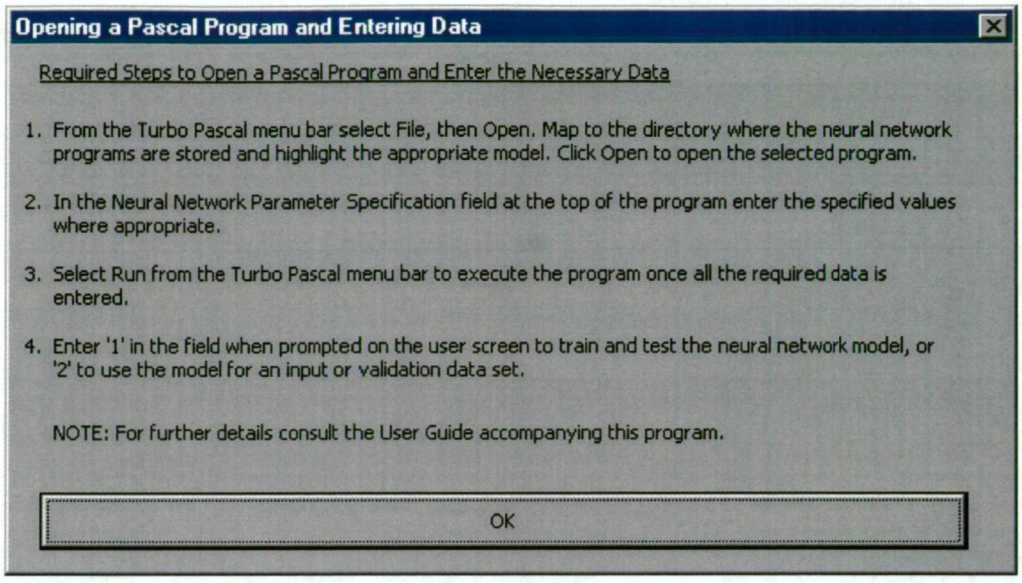


Fig. D.1.15. Message Box Displayed on User Interface Detailing Required Steps to Open a Pascal Program and Enter the Necessary Data

The Turbo Pascal editor screen, as shown in Figure D.1.16, is used to enter the specified values of the neural network constants. It is important to note that unless a predictive or casual importance analysis is being completed the *ImportAnal* and *ParamIndex* fields are left blank. However, if a predictive or casual importance analysis is being completed the letters *pi* or *ci*, respectively, should be entered into the *ImportAnal* field. Further, for a predictive or casual importance analysis the number of the parameter omitted or varied, respectively, in the data sets must be entered into the *ParamIndex* field. Referring to Figure D.1.16, *ImportAnal* field displays 'pi', while the *ParamIndex* field displays '1'. This represents a predictive importance analysis where the first input variable is omitted from the data sets. For each subsequent parameter omitted the *ParamIndex* field is increased accordingly until all parameters have been accounted for. Hence, for n parameters the *ParamIndex* field is subsequently increased from 1 to n. Likewise, the *ParamIndex* field must be changed appropriately for each casual importance analysis. In addition, when training and testing the neural network where any non-contributing input parameters have been omitted from the data sets, the letters *non* should be entered into the *ParamIndex* field and the *NumInputs* field updated accordingly to represent the new number of input variables. In addition, it is important to note that a back-slash (\) is required as the last entry in the *DataDirectory* field, as shown in Figure D.1.16.

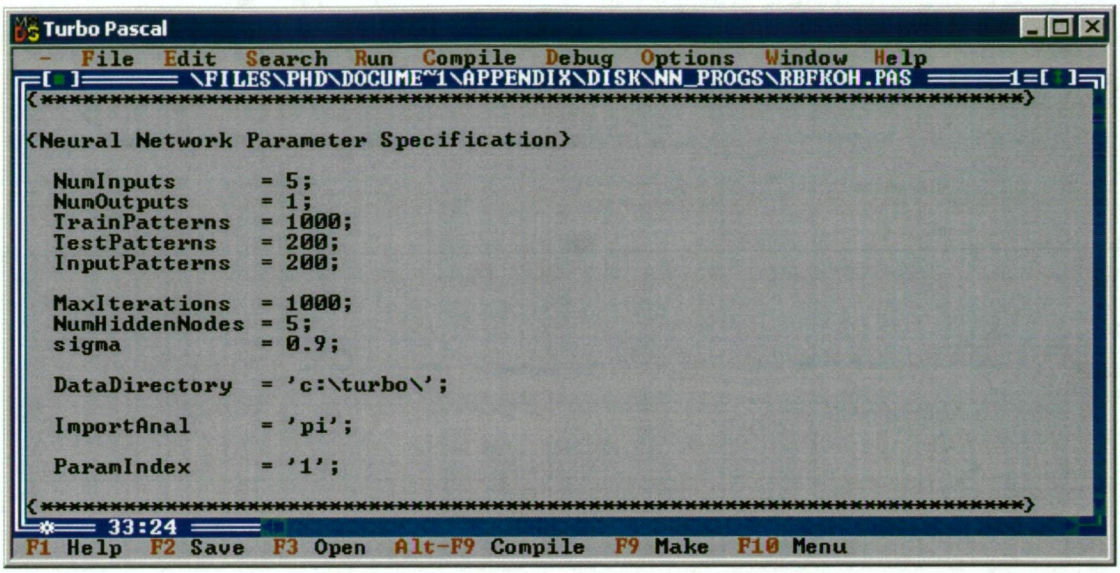


Fig. D.1.16. Illustration of Neural Network Constant Specification in Turbo Pascal Program

View Neural Network Weights File - Select this option to map to the appropriate directory and open the required file containing the neural network weights. If *OK* is selected on the message box shown in Figure D.1.17(a) the *Open* dialog box, shown in Figure D.1.17(b), is presented on the user interface. Use the *Open* dialog box to open the appropriate file.

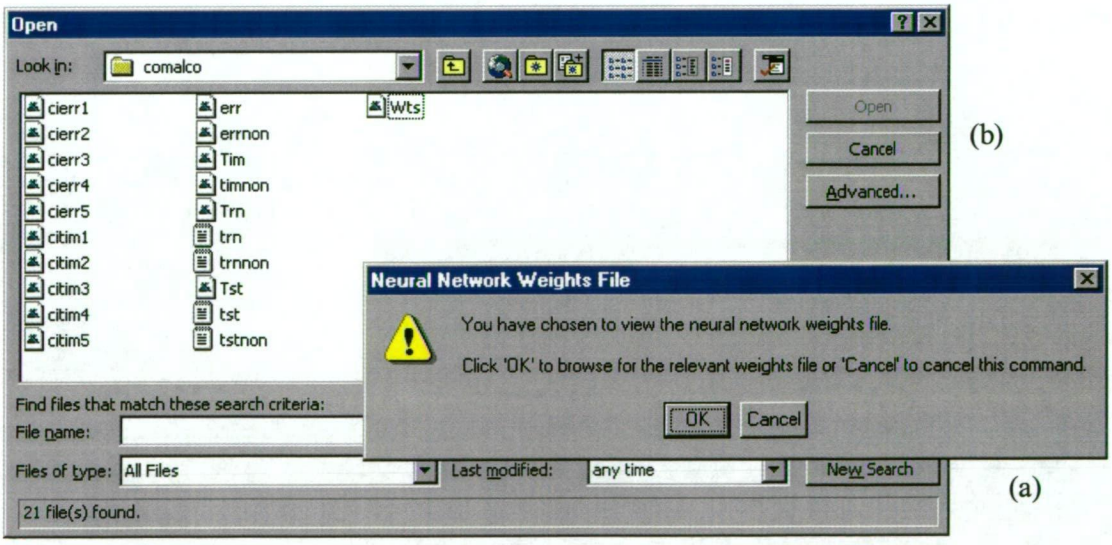


Fig. D.1.17. Message Box Displayed on User Interface (a) Prompting Further Action to Browse for Required File, and (b) Open Dialog Box Required for Directory and File Location

View Neural Network Error File - Select this option to map to the appropriate directory and open the required file containing the neural network training and test error. The *Open* dialog box, shown in Figure D.1.17(b), is presented to the user if *OK* is selected on the message box displayed on the user interface. Use the *Open* dialog box to open the appropriate file.

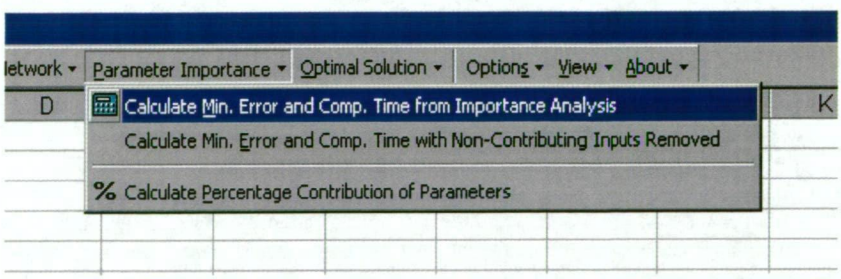
View Neural Network Training Results - Select this option to map to the appropriate directory and open the required file containing the actual and neural network predicted training values. The *Open* dialog box, shown in Figure D.1.17(b), is presented to the user if *OK* is selected on the message box displayed on the user interface. Use the *Open* dialog box to open the appropriate file.

View Neural Network Test Results - Select this option to map to the appropriate directory and open the required file containing the actual and neural network predicted test values. The *Open* dialog box, shown in Figure D.1.17(b), is presented to the user if *OK* is selected on the message box displayed on the user interface. Use the *Open* dialog box to open the appropriate file.

View Neural Network Computation Time - Select this option to map to the appropriate directory and open the required file containing the neural network computation time. The *Open* dialog box, shown in Figure D.1.17(b), is presented to the user if *OK* is selected on the message box displayed on the user interface. Use the *Open* dialog box to open the appropriate file.

Parameter Importance▾

The menu items associated with this drop-down menu are used to determine the minimum train and test error, computation time and number of iterations associated with an analysis. Further, the input parameters can be ranked from highest to lowest importance with the percentage contribution of each input parameter displayed on the user interface.



Calculate Min. Error and Comp. Time from Importance Analysis - Select this menu item to enter the minimum train and test error associated with the original train and test data sets and further, the predictive or casual importance text files. In addition, the computation time and number of iterations associated with each minimum error entry is displayed on the worksheet. The user is required, via the message box shown in Figure D.1.18(a), to select the importance analysis results to be used, either predictive or casual. Further, the user is required, via the message box shown in Figure D.1.18(b), to select whether manual or automatic data entry is desired.

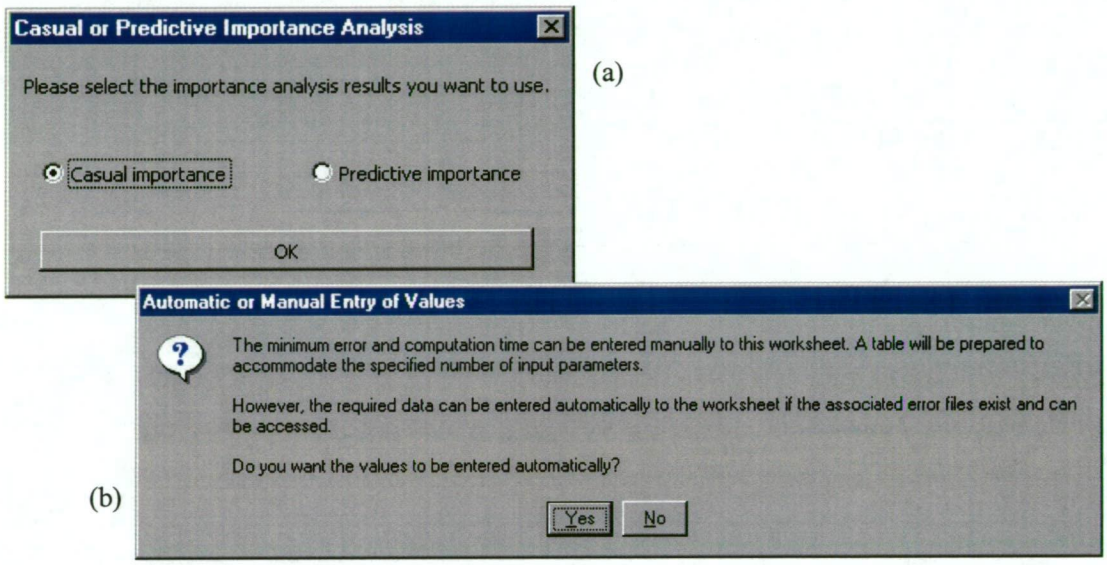


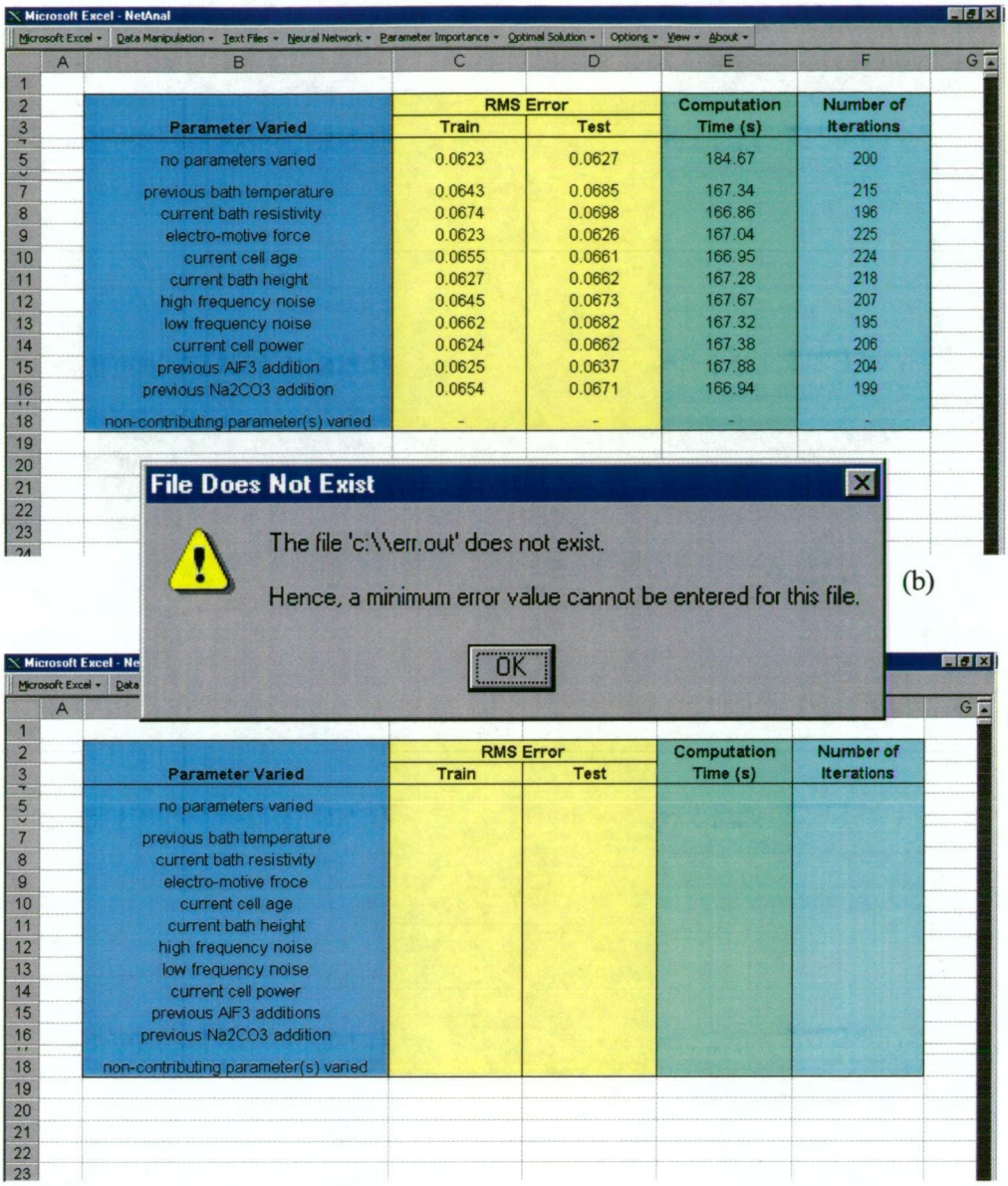
Fig. D.1.18. Message Box Displayed on User Interface (a) Requesting Casual or Predictive Importance Results, and (b) Manual or Automatic Data Entry

Automatically (Yes) - If an importance analysis has been completed and the relevant error files exist and are accessible, select *Yes* for automatic entry of the minimum error, computation time and number of iterations to the worksheet. The required error and computation time files have the identifiers 'err.out' and 'tim.out', respectively, for the original train and test data sets. For the predictive importance analysis results the required error and computation time files have the identifiers 'pierr1.out' and 'pitim1.out', 'pierr2.out' and 'pitim2.out', ..., 'pierr*n*.out' and 'pitim*n*.out', where $n = 1, ..., i$, and i is the number of input parameters. For the casual importance analysis results the required error and computation time files have the identifiers 'cierr1.out' and 'citim1.out', 'cierr2.out' and 'citim2.out', ..., 'cierr*n*.out' and 'citim*n*.out'. If *Yes* is selected the parameter names are automatically entered to the data table, providing

they have been specified previously, else, the identifiers 'input no. 1', 'input no. 2', ..., 'input no. *n*' are entered. The relevant files are automatically opened and the minimum test error is identified. Consequently, the train error, computation time and number of iterations associated with this minimum test error and the minimum test error are copied to the data table, as shown in Figure D.1.19(a). However, if some or all of the required files are not available or accessible, the user is notified using a message box of the type shown in Figure D.1.19(b) and the relevant cells are left blank on the worksheet. Data can be entered into the blank cells manually or automatically if the missing files are created.

If casual importance is selected, *Parameter Varied* is entered at the top of the left-hand side column, while *Parameter Omitted* is entered if predictive importance is selected. The identifier *no parameters omitted* or *no parameters varied* represents the original train and test data sets where all potential input parameters are used, while the parameter name is used to identify the parameter that is omitted or varied in each train and test data set. As a result of this command the *non-contributing parameter(s) omitted* or *non-contributing parameter(s) varied* field is left blank as the error and computation time files required to establish these values are not produced until after this command is complete. Hence, the required data is not available to determine the minimum error, computation time and number of iterations for the non-contributing inputs omitted or varied. However, these values are determined using the menu item introduced in the following section. Nevertheless, if there are no non-contributing input parameters identified in the model then the minimum error, computation time and number of iterations for the non-contributing inputs omitted or varied is the same as that for the original train and test data sets and is documented on the table accordingly.

Manually (No) - Select *No* to create a data table suitable for the number of input parameters associated with the application whereby the minimum error, computation time and number of iterations can be entered manually. The table shown in Figure D.1.19(c) is displayed to the user interface.



Parameter Varied	RMS Error		Computation Time (s)	Number of Iterations
	Train	Test		
no parameters varied	0.0623	0.0627	184.67	200
previous bath temperature	0.0643	0.0685	167.34	215
current bath resistivity	0.0674	0.0698	166.86	196
electro-motive force	0.0623	0.0626	167.04	225
current cell age	0.0655	0.0661	166.95	224
current bath height	0.0627	0.0662	167.28	218
high frequency noise	0.0645	0.0673	167.67	207
low frequency noise	0.0662	0.0682	167.32	195
current cell power	0.0624	0.0662	167.38	206
previous AlF3 addition	0.0625	0.0637	167.88	204
previous Na2CO3 addition	0.0654	0.0671	166.94	199
non-contributing parameter(s) varied	-	-	-	-

Fig. D.1.19. Minimum Error and Computation Time (a) Automatic Data Entry, (b) Missing File Notification, and (c) Manual Data Entry

Calculate Min. Error and Comp. Time with Non-Contributing Inputs Removed - Select this menu item to enter the minimum train and test error associated with the data sets with any non-contributing input parameters removed. The user has the option of entering the values manually or automatically using the message box shown in Figure D.1.18(b).

Automatically (Yes) - If the neural network has been trained and tested using the train and test data sets with the non-contributing input parameters removed, select *Yes* for automatic entry of the minimum error, computation time and number of iterations

onto the worksheet. The required error and computation time files have the identifiers 'errnon.out' and 'timnon.out' for the train and test data sets, respectively. If the required files are not available or accessible, the user is notified and the relevant fields are left blank, as shown in Figure D.1.19(a). However, if the files do exist the relevant values are entered in the respective cells on worksheet, as shown in Figure D.1.20.

Manually (No) - Select *No* to enter the minimum error, computation time and number of iterations manually to the worksheet featured in Figure D.1.19(a).

Microsoft Excel - NetAnal

Microsoft Excel | Data Manipulation | Text Files | Neural Network | Parameter Importance | Optimal Solution | Options | View | About

	Parameter Varied	RMS Error		Computation Time (s)	Number of Iterations
		Train	Test		
5	no parameters varied	0.0623	0.0627	184.67	200
7	previous bath temperature	0.0643	0.0685	167.34	215
8	current bath resistivity	0.0674	0.0698	166.86	196
9	electro-motive force	0.0623	0.0626	167.04	225
10	current cell age	0.0655	0.0661	166.95	224
11	current bath height	0.0627	0.0662	167.28	218
12	high frequency noise	0.0645	0.0673	167.67	207
13	low frequency noise	0.0662	0.0682	167.32	195
14	current cell power	0.0624	0.0662	167.38	206
15	previous AlF3 addition	0.0625	0.0637	167.88	204
16	previous Na2CO3 addition	0.0654	0.0671	166.94	199
18	non-contributing parameter(s) varied	0.0623	0.0626	166.89	201

Fig. D.1.20. Minimum Error and Computation Time with Non-Contributing Inputs Removed Displayed on User Interface

Calculate Percentage Contribution of Parameters - Select this menu item to rank the input parameters in the neural network model from highest to lowest importance. As a result, the percentage contribution of each input parameter is displayed on the user interface. In addition, it can be seen from Figure D.1.21(a) that the total number of contributing input parameters is displayed on the user interface, and also the sum of the percentage contribution of the input parameters, which should always be equivalent to 100.0%.

The percentage contribution of the input parameters is based on the minimum test error from the predictive or casual importance analysis. Hence, the minimum test error must be entered on the relevant worksheet before this command can proceed. A notification is displayed on the user interface if the required data is not available, as shown in Figure D.1.21(b).

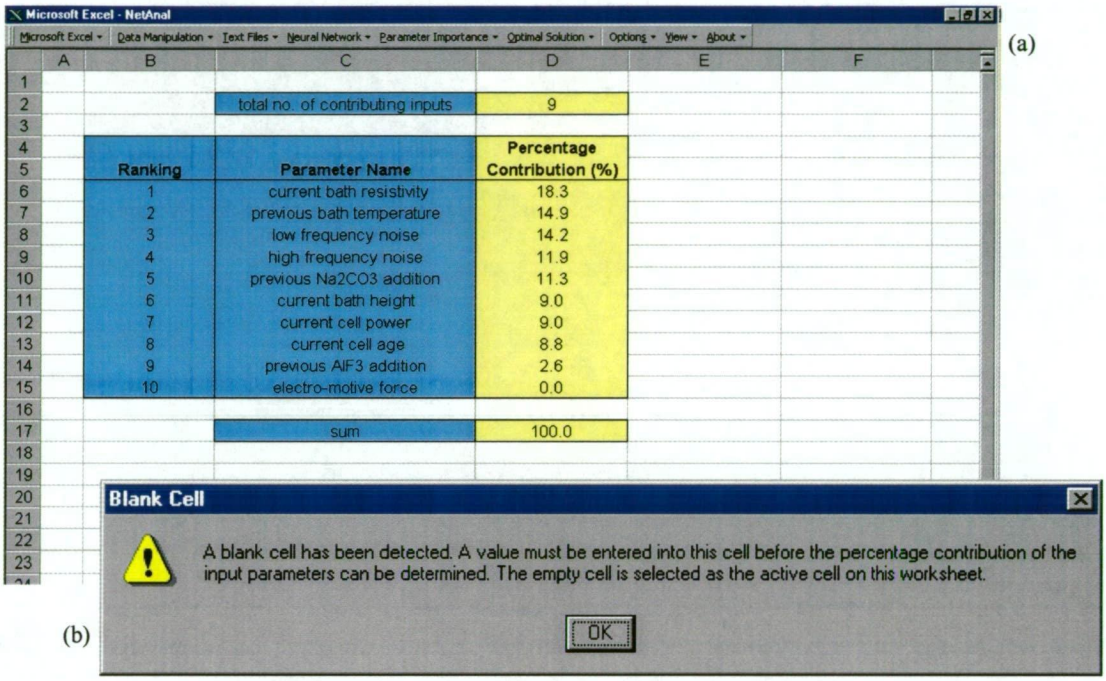
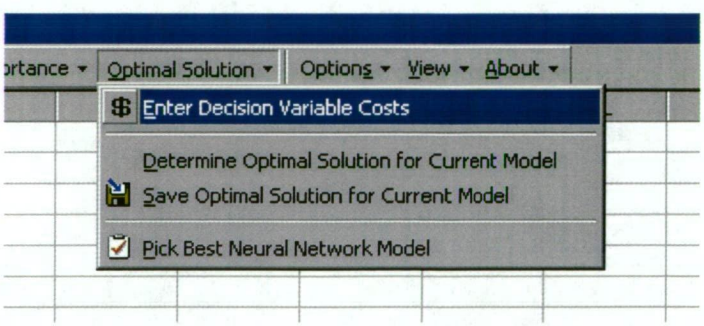


Fig. D.1.21. (a) Percentage Contribution of Input Parameters Displayed on User Interface, and (b) Missing Data Notification

Optimal Solution ▾



The menu items associated with this drop-down menu are used to determine an optimal solution for a neural network model and subsequently, select from the available models the most economically viable neural network to use for an application in order to maximise profitability.

Enter Decision Variable Costs - Select this menu item to create a data table whereby the cost associated with each decision variable can be entered. If the parameter names

have been previously specified, the input parameter names are automatically entered to the worksheet, else, the identifiers 'input no. 1', 'input no. 2', ..., 'input no. *n*' are entered, where $n = 1, \ldots, i$, and i is the number of input parameters. The message box shown in Figure D.1.22(a) is displayed on the user interface informing the user of the decision variable allocation, while the data table shown in Figure D.1.22(b) is displayed on the user interface, ready for entry of the decision variable costs.

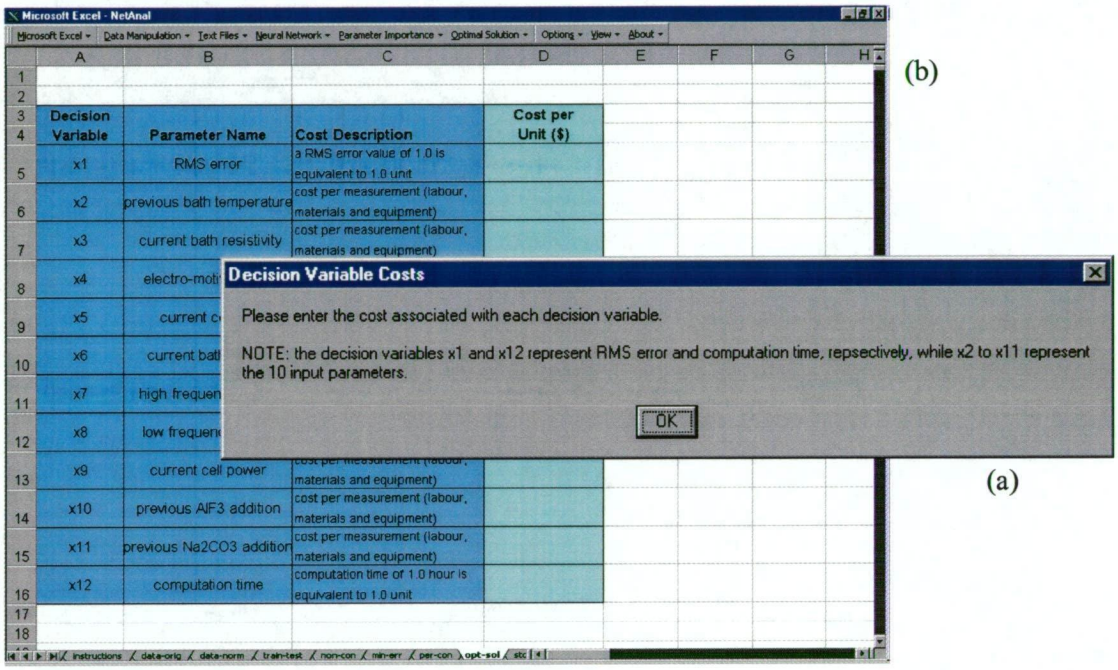


Fig. D.1.22. Decision Variable (a) Allocation Notification, and (b) Cost Data Entry Table

Determine Optimal Solution for Current Model - Select this menu item to determine the optimal solution for a neural network model. A prerequisite for this command is entry of the minimum test error and computation time associated with the original analysis, predictive importance analysis and analysis with any non-contributing inputs removed. It is necessary that the predictive importance analysis results be used, rather than the casual importance analysis results, as the decrease in computation time associated with the removal of an input parameter is calculated during the predictive importance analysis, but not during the casual importance analysis. Further, while the predictive and casual importance analyses typically yield similar results it is more accurate to use the predictive importance analysis results as it involves the removal of input parameters from the model, which is the objective of the optimisation procedure. In addition, the decision variable costs must be entered

before the optimal solution can be determined for the current model. If the necessary conditions are satisfied, a procedure is used to ensure this, the user is requested, via a message box displayed on the user interface, to confirm the decision variable costs are correct, as shown in Figure D.1.23(a), and the minimum error and computation time values are correct, as shown in Figure D.1.23(b).

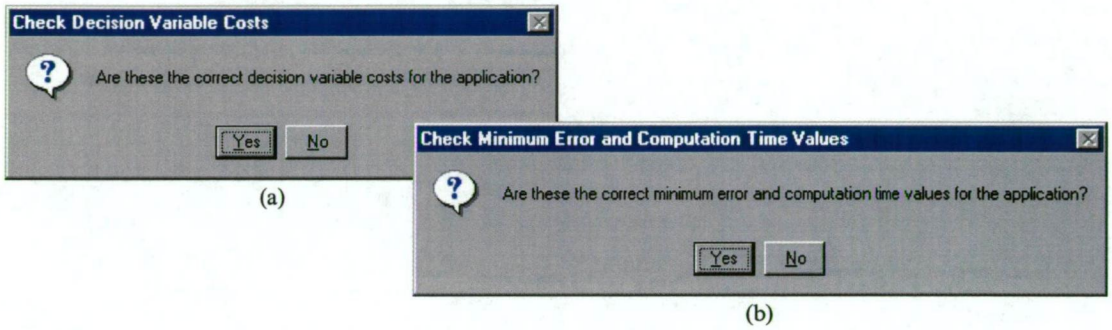


Fig. D.1.23. Message Boxes Displayed on User Interface Requesting Confirmation of (a) Decision Variable Costs, and (b) Minimum Error and Computation Time Values

If the decision variable costs and minimum error and computation time values are confirmed, the optimal solution is established and displayed on the user interface, as shown in Figure D.1.24(a). Further, the result of the analysis is displayed on the user interface, via a message box, as shown in Figure D.1.24(b). In addition, the original solution is displayed on the worksheet for comparison.

Save Optimal Solution for Current Model - This menu item is selected if there are multiple models considered for an application. The execution of this command stores the current optimal solution to a separate worksheet that may contain multiple optimal solutions. The user is prompted, using the message box shown in Figure D.1.25(a), to clear the contents of the worksheet where the optimal solution will be stored. Select *Yes* on the message box if previously stored optimal solutions are no longer required. An identifier for the current optimal solution is requested using a message box displayed on the user interface, as shown in Figure D.1.25(b), while the model identifier is confirmed, as shown in Figure D.1.25(c), before proceeding.

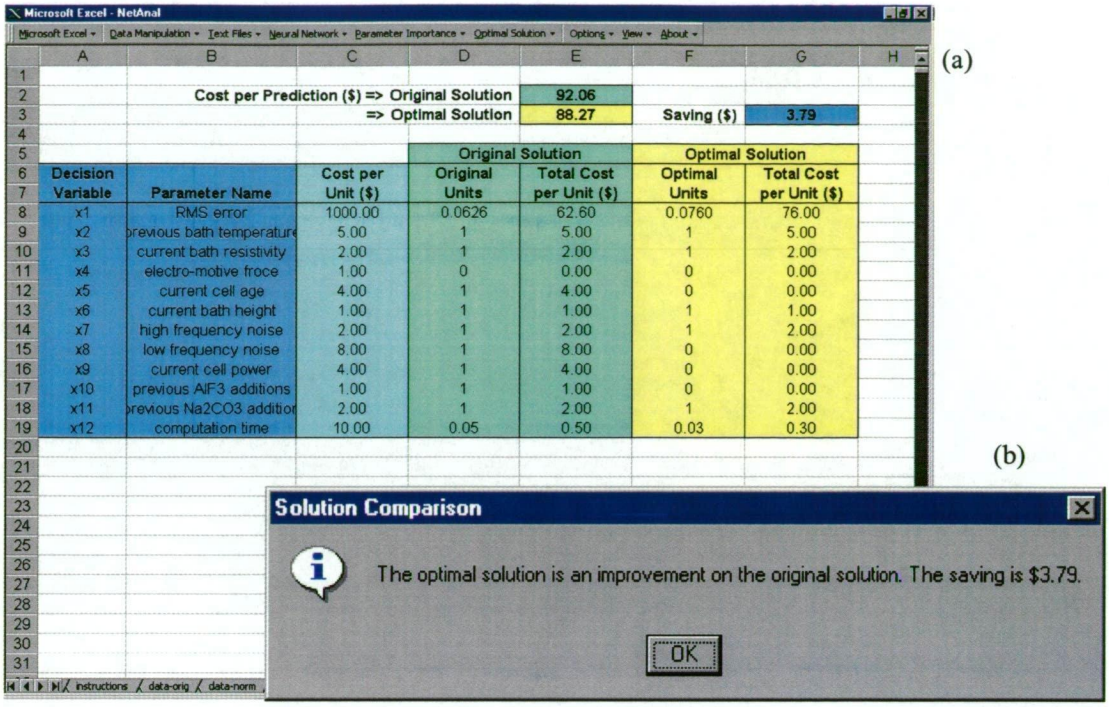


Fig. D.1.24. (a) Original and Optimal Solutions, and (b) Message Box Highlighting Result of Analysis

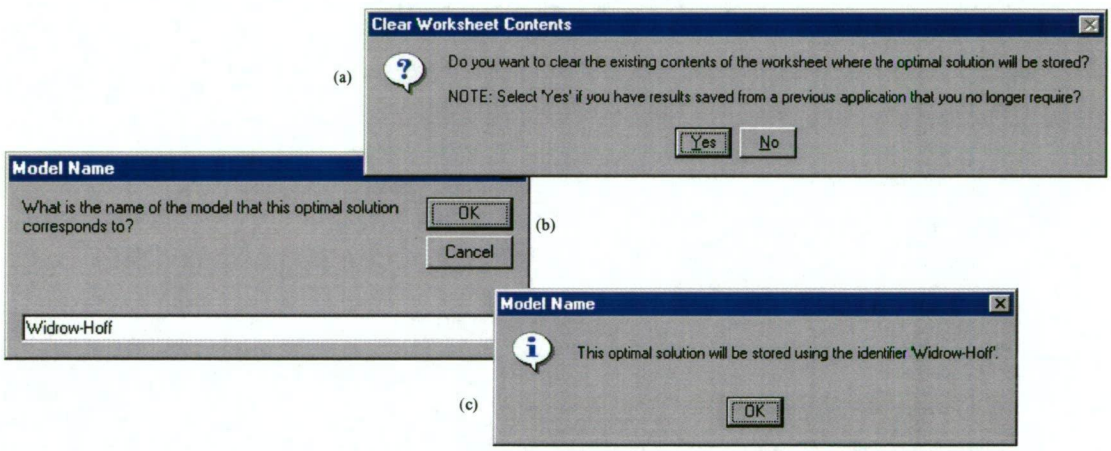
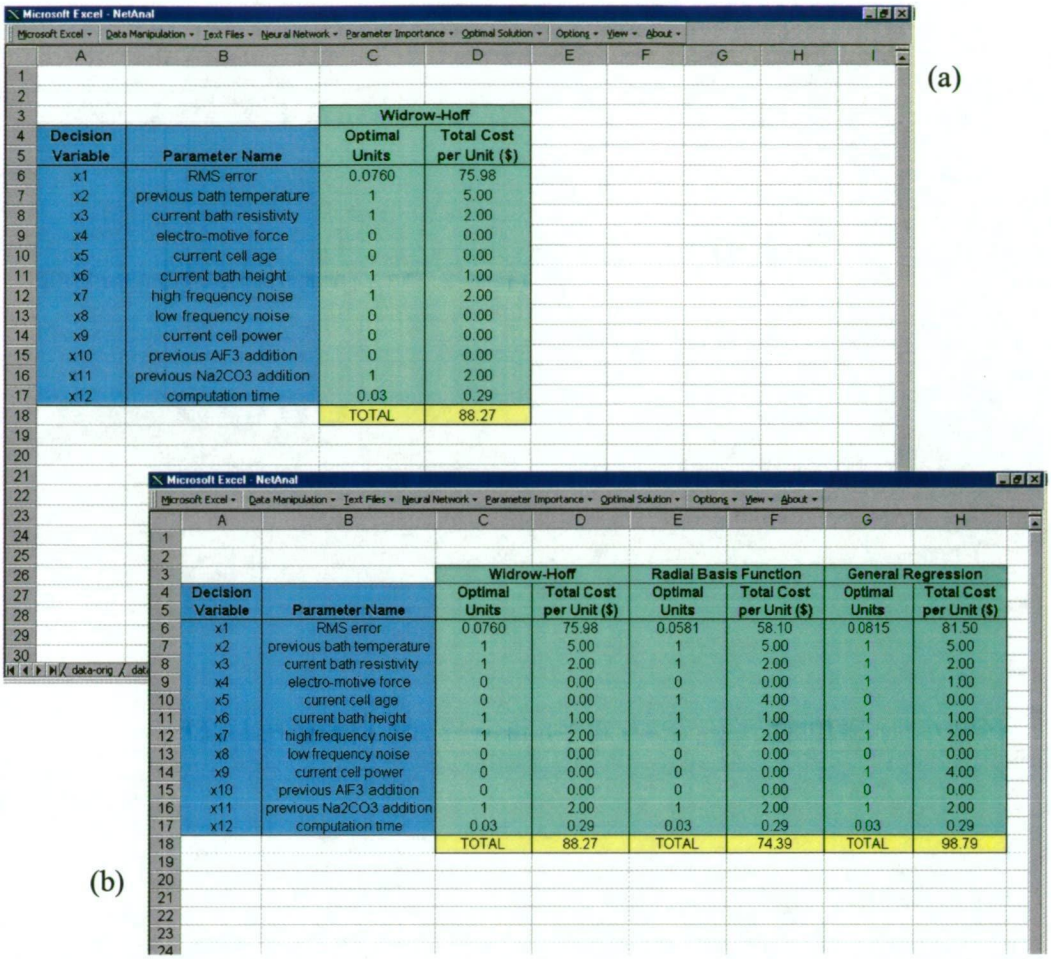


Fig. D.1.25. Message Boxes Displayed on user Interface (a) Prompting Clear Contents, (b) Requesting Model Identifier, and (c) Confirming Model Identifier

The optimal solution is saved to a separate worksheet and displayed on the user interface as shown in Figure D.1.26(a). Subsequently, if further optimal solutions are established for different models, the solution can be saved to this worksheet, as shown in Figure D.1.26(b). Hence, for any application, an optimal solution can be determined for multiple models and the results conveniently saved, ready for model selection.



(a)

Microsoft Excel - NetAnal

Microsoft Excel | Data Manipulation | Text Files | Neural Network | Parameter Importance | Optimal Solution | Options | View | About

Decision Variable	Parameter Name	Widrow-Hoff Optimal Units	Widrow-Hoff Total Cost per Unit ($)
x1	RMS error	0.0760	75.98
x2	previous bath temperature	1	5.00
x3	current bath resistivity	1	2.00
x4	electro-motive force	0	0.00
x5	current cell age	0	0.00
x6	current bath height	1	1.00
x7	high frequency noise	1	2.00
x8	low frequency noise	0	0.00
x9	current cell power	0	0.00
x10	previous AlF3 addition	0	0.00
x11	previous Na2CO3 addition	1	2.00
x12	computation time	0.03	0.29
		TOTAL	88.27

(b)

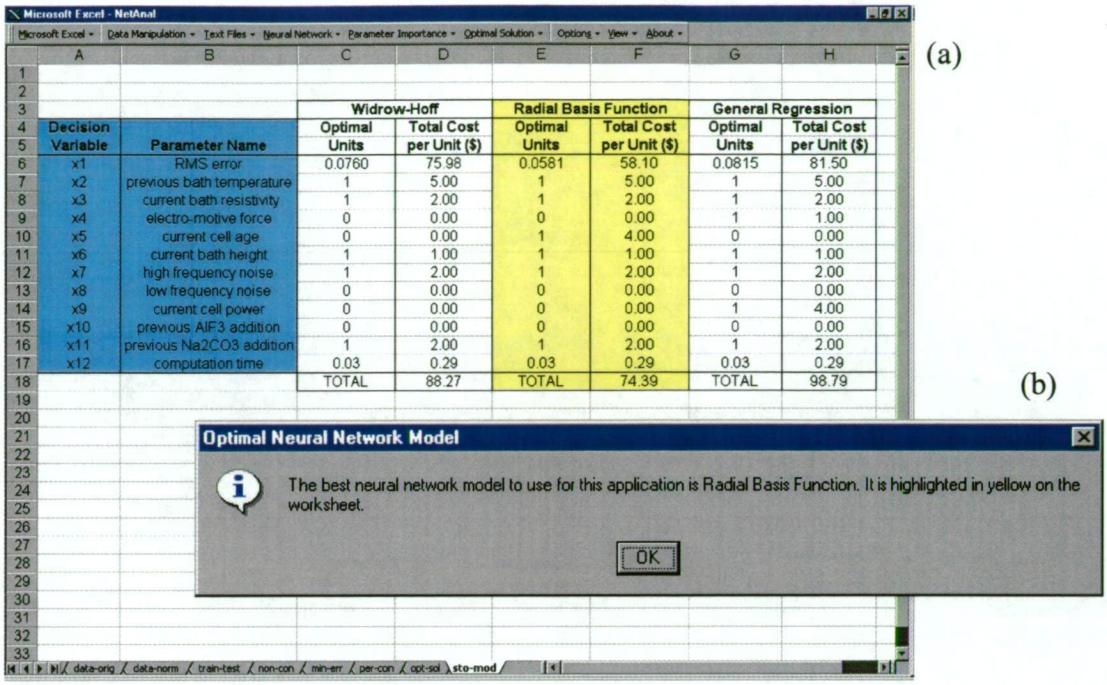
Microsoft Excel - NetAnal

Microsoft Excel | Data Manipulation | Text Files | Neural Network | Parameter Importance | Optimal Solution | Options | View | About

Decision Variable	Parameter Name	Widrow-Hoff Optimal Units	Widrow-Hoff Total Cost per Unit ($)	Radial Basis Function Optimal Units	Radial Basis Function Total Cost per Unit ($)	General Regression Optimal Units	General Regression Total Cost per Unit ($)
x1	RMS error	0.0760	75.98	0.0581	58.10	0.0815	81.50
x2	previous bath temperature	1	5.00	1	5.00	1	5.00
x3	current bath resistivity	1	2.00	1	2.00	1	2.00
x4	electro-motive force	0	0.00	0	0.00	1	1.00
x5	current cell age	0	0.00	1	4.00	0	0.00
x6	current bath height	1	1.00	1	1.00	1	1.00
x7	high frequency noise	1	2.00	1	2.00	1	2.00
x8	low frequency noise	0	0.00	0	0.00	0	0.00
x9	current cell power	0	0.00	0	0.00	1	4.00
x10	previous AlF3 addition	0	0.00	0	0.00	0	0.00
x11	previous Na2CO3 addition	1	2.00	1	2.00	1	2.00
x12	computation time	0.03	0.29	0.03	0.29	0.03	0.29
		TOTAL	88.27	TOTAL	74.39	TOTAL	98.79

Fig. D.1.26. Optimal Solution Saved and Displayed on User Interface (a) Single Model, and (b) Multiple Models

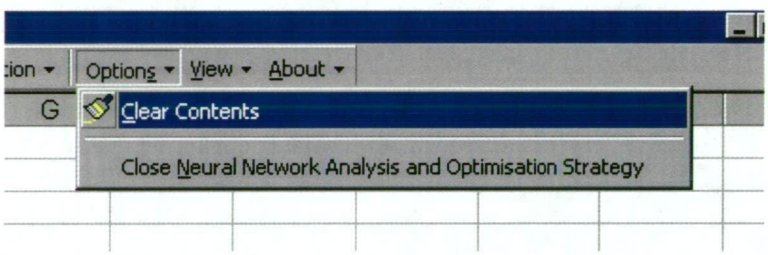
Pick Best Neural Network Model - This menu item is used to select the most economically viable neural network model to use for a specific application. This analysis is completed on multiple optimal solutions, Figure D.1.26(b) is an example of three optimal solutions. However, if only a single optimal solution exists and this command is executed, the single optimal solution is highlighted as the best model to use for the application. The best neural network model is selected by comparing the total cost associated with each of the available optimal solutions, highlighting the solution with minimum cost, as shown in Figure D.1.27(a). A message is displayed on the user interface notifying the user that the best model to use for the application has been selected, as shown in Figure D.1.27(b).



			Widrow-Hoff		Radial Basis Function		General Regression	
Decision Variable	Parameter Name		Optimal Units	Total Cost per Unit ($)	Optimal Units	Total Cost per Unit ($)	Optimal Units	Total Cost per Unit ($)
x1	RMS error		0.0760	75.98	0.0581	58.10	0.0815	81.50
x2	previous bath temperature		1	5.00	1	5.00	1	5.00
x3	current bath resistivity		1	2.00	1	2.00	1	2.00
x4	electro-motive force		0	0.00	0	0.00	1	1.00
x5	current cell age		0	0.00	1	4.00	0	0.00
x6	current bath height		1	1.00	1	1.00	1	1.00
x7	high frequency noise		1	2.00	1	2.00	1	2.00
x8	low frequency noise		0	0.00	0	0.00	0	0.00
x9	current cell power		0	0.00	0	0.00	1	4.00
x10	previous AlF3 addition		0	0.00	0	0.00	0	0.00
x11	previous Na2CO3 addition		1	2.00	1	2.00	1	2.00
x12	computation time		0.03	0.29	0.03	0.29	0.03	0.29
			TOTAL	88.27	TOTAL	74.39	TOTAL	98.79

Fig. D.1.27. (a) Selection of Best Neural Network Model, and (b) Notification that Best Neural Network Model has been Selected

Options ▾



The menu items associated with this particular drop-down menu include the option to clear the contents of a worksheet or close the *Neural Network Analysis and Optimisation Strategy* program.

Clear Contents - Select this option to remove the existing contents of the active, or displayed, worksheet, including all values, formulae, patterns and borders. However, the contents of the *introduction* and *instructions* worksheets cannot be cleared. These worksheets are protected from this action. A notification message is displayed to the user interface if this menu item is selected while the *introduction* or *instructions* worksheet is active, as shown in Figure D.1.28(a). In addition, to prevent accidentally removing the valuable contents of a worksheet a notification message, shown in Figure D.1.28(b) is displayed to the user interface to ensure this action is desired.

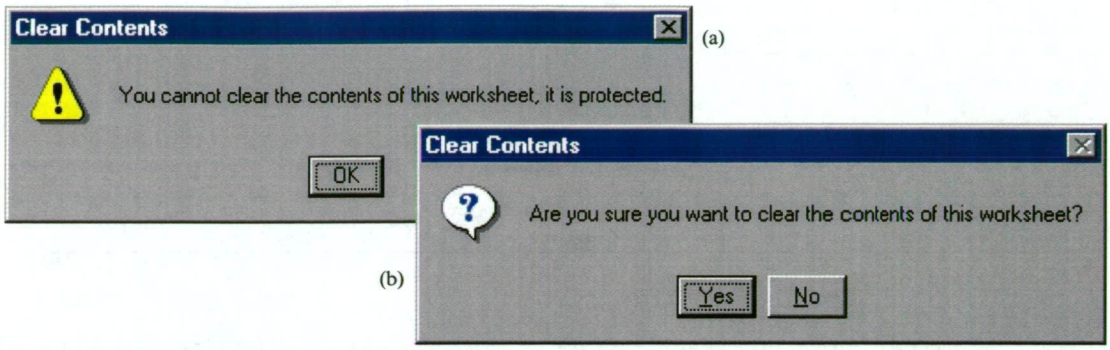


Figure D.1.28. Message Boxes (a) Notification that Worksheet is Protected, and (b) Requesting Confirmation to Clear the Contents of the Active Worksheet

Close Neural Network Analysis and Optimisation Strategy - Select this option to finish using the program. The message box shown in Figure D.1.29 is displayed on the user screen before the program closes.

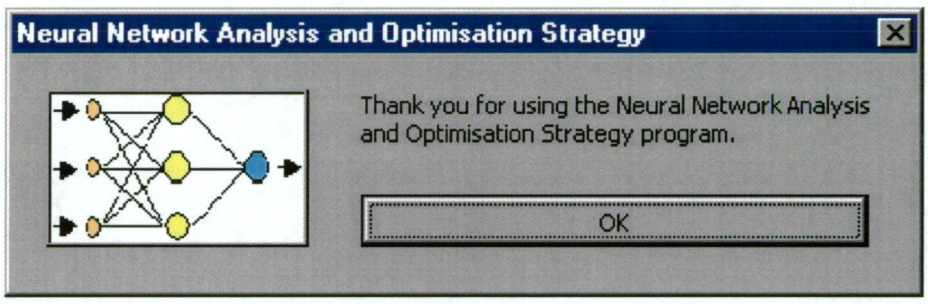
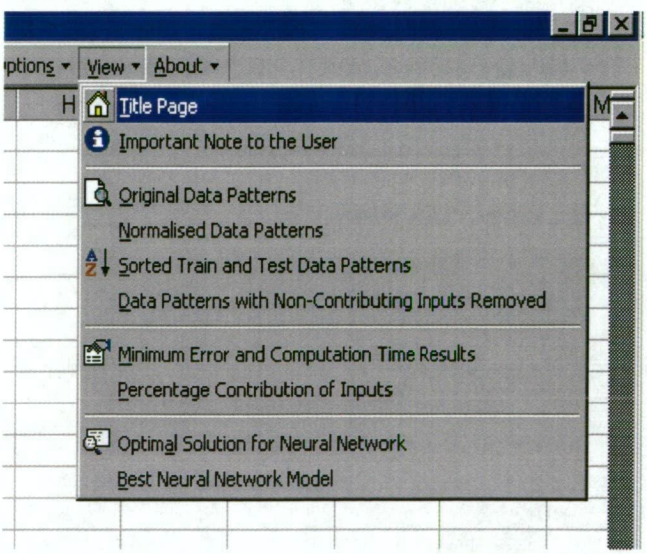


Fig. D.1.29. *Neural Network Analysis and Optimisation Strategy* Program Close Message Box Displayed on User Interface

View▾

The menu items associated with this particular drop-down menu direct the user to the specified worksheet. However, there is no procedure used to check the content of a worksheet prior to displaying it. Hence, the selected worksheet is displayed on the user interface regardless of its content. Therefore, a blank



worksheet may be displayed on the user interface if no prior data has been entered to the selected worksheet.

Title Page - Selecting this option activates the *introduction* worksheet, containing the program title and neural network graphic.

Important Note to the User - Selecting this option activates the *instructions* worksheet, containing program overview and copyright notice.

Original Data Patterns - Selecting this option activates the *data-orig* worksheet. The parameter names, original data patterns and minimum and maximum values of the original data patterns are displayed on this worksheet.

Normalised Data Patterns - Selecting this option activates the *data-norm* worksheet. The parameter names, normalised data patterns and minimum and maximum values of the normalised data patterns are displayed on this worksheet.

Sorted Train and Test Data Patterns - Selecting this option activates the *train-test* worksheet. The train and test data sets, using the normalised data patterns, are displayed on the worksheet.

Data Patterns with Non-Contributing Inputs Removed - Selecting this option activates the *non-con* worksheet. The train and test data sets with any non-contributing input parameters removed, using the normalised data patterns, are displayed on the worksheet.

Minimum Error and Computation Time Results - Selecting this option activates the *min-err* worksheet. The minimum train and test error from the train and test data sets, importance analysis and train and test data sets with any non-contributing input parameters omitted are displayed on the worksheet. Further, the computation time associated with the minimum train and test error and the relevant iteration number at which the minimum test error occurred is displayed on the worksheet.

Percentage Contribution of Inputs - Selecting this option activates the *per-con* worksheet. The percentage contribution of each input parameter is displayed on the worksheet.

Optimal Solution for Neural Network - Selecting this option activates the *opt-sol* worksheet. The original and optimal solution for the current neural network model is displayed on the worksheet.

Best Neural Network Model - Selecting this option activates the *sto-mod* worksheet. The original and optimal solutions for any number of neural network models are displayed on the worksheet.

About▾

The *About* drop-down menu contains a single menu item, *About Neural Network Analysis and Optimisation Strategy*.

About Neural Network Analysis and Optimisation Strategy - Select this menu item to view a message box containing author details and copyright information, as shown in Figure D.1.30.

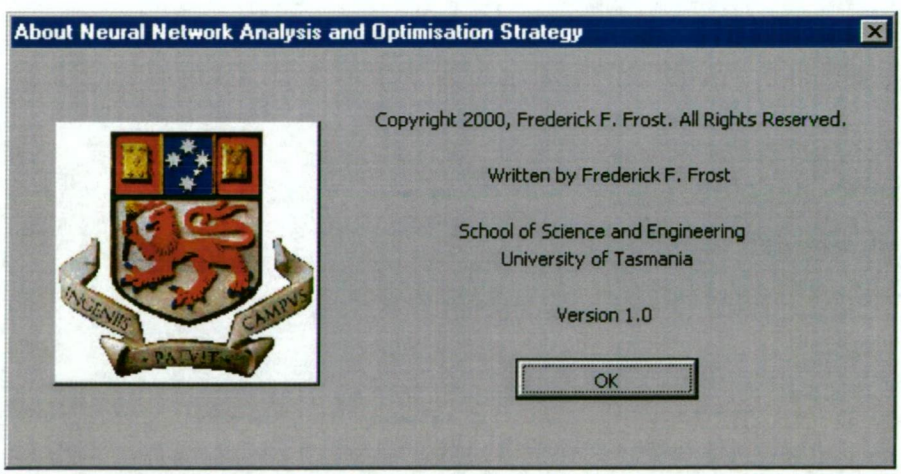


Fig. D.1.30. About *Neural Network Analysis and Optimisation Strategy* Message Box Displayed on User Interface

D.1.4 Error Trapping and Handling in Neural Network Analysis and Optimisation Strategy

While the potential for a run-time error to occur during code execution has been minimised through use of specific programming techniques and rigorous testing of the associated code, such errors have the opportunity to occur due to undesirable actions initiated by the program user, such as attempting to open a file that does not exist. To prevent an unexpected run-time error stopping the program code prematurely, potentially leaving data in an unpredictable state, specific error handlers have been incorporated into the program code of the *Neural Network Analysis and Optimisation Strategy*. For instance, if a file is referenced, but in fact does not exist, the code is designed to notify the user of the non-existence and continue with the subsequent command in the program code. Similarly, specific data verification techniques are used to ensure user data entry is correct. For example, an integer must be entered when an integer value is required, characters outside the range 0 to 9 are not accepted. Further, the 'On Error' function available with Microsoft® Visual Basic™ is utilised to ensure that program code is not halted unexpectedly during execution.

In addition, appropriate verification procedures have been integrated into the programming code to ensure any necessary prerequisites are satisfied before specific commands are executed. For example, the train and test data sets cannot be developed if the data patterns have not been entered. Appropriate message and custom dialogue boxes are used to notify the user of any prerequisites required for selected commands to proceed. Any prerequisites required for each available model analysis and optimisation command are detailed in the preceding documentation.

D.1.5 Saving Your Work

While the *Neural Network Analysis and Optimisation Strategy* program is distributed in read-only format, saving necessary modifications to the workbook is permitted provided the workbook is saved using an identifier other than *NetAnal.xls* or in a different location. Hence, to save changes to *NetAnal.xls* select, from the *NeuralNet* toolbar, *Microsoft Excel*, *File*, *Save* or *SaveAs*, map to the appropriate directory or location, specify the file name and click *Save*. Consequently, the modified workbook will be saved in the specified location.

Alternatively, rather than saving the entire workbook, the contents of a worksheet can be saved by copying the required data from *NetAnal.xls* and pasting the copied data in a new workbook. Likewise, select *Save* or *SaveAs* to save the contents of the new workbook to a specified location. Multiple worksheets can be saved using this technique.

D.2 NEURAL NETWORK ANALYSIS AND OPTIMISATION STRATEGY PROGRAM SOURCE CODE

Formatted descriptive source code is available on the accompanying software for the *Neural Network Analysis and Optimisation Strategy* program. The source code is available in the directory *Opt_Prog* in the sub-directory *Sce_Code*. The source code file has the identifier *Opt_Code.doc*.

APPENDIX E

Measurement Cost of Parameters Used in Industrial Application Modelling

TABLE E.1. Measurement Cost Associated with Process Parameters Used for Industrial Application Modelling

Process Parameter	Cost Component per Reduction Cell	Component Cost per Measurement ($)	Total Cost per Measurement ($)
target bath temperature	n/a (specified value)	0.00	0.00
bath temperature	labour (manual)	0.52	
	thermocouple consumption	0.61	
	associated injuries	0.02	1.15
bath height	labour (manual)	0.52	
	steel rod consumption	0.01	
	associated injuries	0.02	0.55
bath resistivity	equipment maintenance	0.02	0.02
emf	labour (manual)	0.28	
	equipment maintenance	0.01	0.29
AlF_3 addition	labour (manual)	0.52	
	equipment consumption	0.61	
	associated injuries	0.02	
	equipment maintenance	0.02	1.17
Na_2CO_3 addition	labour (manual)	0.52	
	equipment consumption	0.61	
	associated injuries	0.02	
	equipment maintenance	0.02	1.17
cell power	equipment maintenance	0.04	0.04
cell age	n/a (calculated value)	0.00	0.00
F content	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	4.00	5.03
Na content	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	4.00	5.03
temperature reference	n/a (specified value)	0.00	0.00
low frequency noise	equipment maintenance	0.01	0.01
high frequency noise	equipment maintenance	0.01	0.01
cell resistance	equipment maintenance	0.03	0.03

Fe content	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	2.80	3.83
Si content	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	2.80	3.83
lining voltage drop	labour	0.51	
	equipment consumption	0.15	
	associated injuries	0.02	0.68
AE frequency	equipment maintenance	0.02	0.02
AE duration	equipment maintenance	0.02	0.02
AE energy	equipment maintenance	0.02	0.02
unsched. anode change	labour (manual)	0.26	0.26
rod height	labour (manual)	0.61	0.61
cell voltage	equipment maintenance	0.01	0.01
Fe/V	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	2.80	3.83
Fe/Ga	labour (manual)	0.97	
	equipment consumption	0.06	
	laboratory analysis	2.80	3.83
high temp. exc. count	labour (manual)	0.52	
	thermocouple consumption	0.61	
	associated injuries	0.02	1.15
low temp. exc. count	labour (manual)	0.52	
	thermocouple consumption	0.61	
	associated injuries	0.02	1.15

Neural Network Solutions for Studied Industrial Applications

TABLE F.1. Comparison of Original and Optimal Solutions for WH Neural Network for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.1121	17.82	0.1172	18.64
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	0	0.00
x_4	bath height	0.55	0	0.00	0	0.00
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	0	0.00	0	0.00
x_7	AlF_3 addition (t-1)	1.17	1	1.17	0	0.00
x_8	Na_2CO_3 addition (t-1)	1.17	1	1.17	0	0.00
x_9	cell power	0.04	0	0.00	0	0.00
x_{10}	cell age	0.00	0	0.00	0	0.00
x_{11}	F content	5.03	0	0.00	0	0.00
x_{12}	Na content	5.03	1	5.03	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.08	0.00	0.06	0.00
		TOTAL	-	26.36	-	18.66

TABLE F.2. Comparison of Original and Optimal Solutions for BP1 Neural Network for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.0774	12.31	0.0804	12.78
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	1	1.15
x_4	bath height	0.55	1	0.55	0	0.00
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	0	0.00	0	0.00
x_7	AlF_3 addition (t-1)	1.17	0	0.00	0	0.00
x_8	Na_2CO_3 addition (t-1)	1.17	1	1.17	1	1.17
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	cell age	0.00	1	0.00	1	0.00
x_{11}	F content	5.03	1	5.03	0	0.00
x_{12}	Na content	5.03	0	0.00	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.28	0.00	0.24	0.00
		TOTAL	-	20.27	-	15.16

TABLE F.3. Comparison of Original and Optimal Solutions for BP2 Neural Network for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.0719	11.43	0.0873	13.88
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	1	1.15
x_4	bath height	0.55	1	0.55	1	0.55
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	1	0.29	1	0.29
x_7	AlF_3 addition (t-1)	1.17	1	1.17	1	1.17
x_8	Na_2CO_3 addition (t-1)	1.17	1	1.17	0	0.00
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	cell age	0.00	1	0.00	1	0.00
x_{11}	F content	5.03	1	5.03	0	0.00
x_{12}	Na content	5.03	1	5.03	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.86	0.00	0.77	0.00
		TOTAL	-	25.88	-	17.10

TABLE F.4. Comparison of Original and Optimal Solutions for RBF Neural Network for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.0727	11.56	0.0755	12.00
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	1	1.15
x_4	bath height	0.55	1	0.55	0	0.00
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	0	0.00	0	0.00
x_7	AlF_3 addition (t-1)	1.17	1	1.17	1	1.17
x_8	Na_2CO_3 addition (t-1)	1.17	0	0.00	0	0.00
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	cell age	0.00	1	0.00	1	0.00
x_{11}	F content	5.03	1	5.03	0	0.00
x_{12}	Na content	5.03	1	5.03	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.37	0.00	0.31	0.00
		TOTAL	-	24.55	-	14.38

TABLE F.5. Comparison of Original and Optimal Solutions for RBFKOH Neural Network for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.0715	11.37	0.0723	11.50
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	1	1.15
x_4	bath height	0.55	0	0.00	0	0.00
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	0	0.00	0	0.00
x_7	AlF_3 addition (t-1)	1.17	1	1.17	1	1.17
x_8	Na_2CO_3 addition (t-1)	1.17	0	0.00	0	0.00
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	cell age	0.00	1	0.00	1	0.00

x_{11}	F content	5.03	1	5.03	0	0.00
x_{12}	Na content	5.03	0	0.00	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.75	0.00	0.73	0.00
		TOTAL	-	18.78	-	13.88

TABLE F.6. Comparison of Original and Optimal Solutions for GRNN for Electrolyte Additive Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.0671	10.67	0.0742	11.80
x_2	target bath temperature	0.00	1	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15	0	0.00
x_4	bath height	0.55	0	0.00	0	0.00
x_5	bath resistivity	0.02	1	0.02	1	0.02
x_6	emf	0.29	0	0.00	0	0.00
x_7	AlF_3 addition (t-1)	1.17	1	1.17	0	0.00
x_8	Na_2CO_3 addition (t-1)	1.17	0	0.00	0	0.00
x_9	cell power	0.04	0	0.00	0	0.00
x_{10}	cell age	0.00	0	0.00	0	0.00
x_{11}	F content	5.03	0	0.00	0	0.00
x_{12}	Na content	5.03	0	0.00	0	0.00
x_{13}	temperature reference	0.00	1	0.00	1	0.00
x_{14}	computation time (hr)	0.70E-03	0.01	0.00	0.01	0.00
		TOTAL	-	13.01	-	11.82

TABLE F.7. Original Solution for Multi-Variable Regression Analysis (MVRA) Model for Electrolyte Additive Prediction Application

Decision		Cost per	Original Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)
x_1	RMS error (test)	159.00	0.1129	17.95
x_2	target bath temperature	0.00	1	0.00
x_3	bath temperature	1.15	1	1.15
x_4	bath height	0.55	0	0.00
x_5	bath resistivity	0.02	1	0.02
x_6	emf	0.29	1	0.29
x_7	AlF_3 addition (t-1)	1.17	1	1.17
x_8	Na_2CO_3 addition (t-1)	1.17	0	0.00
x_9	cell power	0.04	1	0.04
x_{10}	cell age	0.00	0	0.00
x_{11}	F content	5.03	0	0.00
x_{12}	Na content	5.03	0	0.00
x_{13}	temperature reference	0.00	0	0.00
x_{14}	computation time (hr)	0.70E-03	0.00	0.00
		TOTAL	-	20.62

TABLE F.8. Comparison of Original and Optimal Solutions for WH Neural Network for Cell Failure Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.0761	38.18	0.0818	41.04
x_2	Fe content (weekly)	3.83	1	3.83	0	0.00

x_3	Si content (weekly)	3.83	1	3.83	0	0.00
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	0	0.00	0	0.00
x_6	bath temperature	1.15	1	1.15	0	0.00
x_7	cell age	0.00	1	0.00	1	0.00
x_8	AE frequency	0.02	1	0.02	1	0.02
x_9	AE duration	0.02	1	0.02	1	0.02
x_{10}	un. anode change	0.26	1	0.26	1	0.26
x_{11}	rod height	0.61	1	0.61	0	0.00
x_{12}	cell power	0.04	1	0.04	1	0.04
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	0	0.00	0	0.00
x_{15}	cell resistance	0.03	1	0.03	1	0.03
x_{16}	emf	0.29	0	0.00	0	0.00
x_{17}	bath resistivity	0.02	1	0.02	1	0.02
x_{18}	cell voltage	0.01	1	0.01	1	0.01
x_{19}	AE energy	0.02	1	0.02	1	0.02
x_{20}	Si content	3.83	1	3.83	0	0.00
x_{21}	Fe content	3.83	1	3.83	0	0.00
x_{22}	Fe/V	3.83	1	3.83	0	0.00
x_{23}	Fe/Ga	3.83	0	0.00	0	0.00
x_{24}	high temp. count	1.15	1	1.15	1	1.15
x_{25}	low temp. count	1.15	1	1.15	0	0.00
x_{26}	computation time (hr)	0.51E-02	0.08	0.00	0.02	0.00
		TOTAL	-	62.50	-	43.30

TABLE F.9. Comparison of Original and Optimal Solutions for BP1 Neural Network for Cell Failure Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.0451	22.63	0.0451	22.63
x_2	Fe content (weekly)	3.83	1	3.83	1	3.83
x_3	Si content (weekly)	3.83	1	3.83	1	3.83
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	1	0.55	1	0.55
x_6	bath temperature	1.15	1	1.15	1	1.15
x_7	cell age	0.00	1	0.00	1	0.00
x_8	AE frequency	0.02	1	0.02	1	0.02
x_9	AE duration	0.02	1	0.02	1	0.02
x_{10}	un. anode change	0.26	1	0.26	1	0.26
x_{11}	rod height	0.61	1	0.61	1	0.61
x_{12}	cell power	0.04	1	0.04	1	0.04
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	1	0.01	1	0.01
x_{15}	cell resistance	0.03	1	0.03	1	0.03
x_{16}	emf	0.29	1	0.29	1	0.29
x_{17}	bath resistivity	0.02	1	0.02	1	0.02
x_{18}	cell voltage	0.01	1	0.01	1	0.01
x_{19}	AE energy	0.02	1	0.02	1	0.02
x_{20}	Si content	3.83	1	3.83	1	3.83
x_{21}	Fe content	3.83	1	3.83	1	3.83
x_{22}	Fe/V	3.83	1	3.83	1	3.83
x_{23}	Fe/Ga	3.83	1	3.83	1	3.83
x_{24}	high temp. count	1.15	1	1.15	1	1.15
x_{25}	low temp. count	1.15	0	0.00	0	0.00
x_{26}	computation time (hr)	0.51E-02	1.23	0.01	1.23	0.01
		TOTAL	-	50.48	-	50.48

TABLE F.10. Comparison of Original and Optimal Solutions for BP2 Neural Network for Cell Failure Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.0580	29.10	0.0789	39.56
x_2	Fe content (weekly)	3.83	1	3.83	1	3.83
x_3	Si content (weekly)	3.83	1	3.83	1	3.83
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	0	0.00	0	0.00
x_6	bath temperature	1.15	1	1.15	0	0.00
x_7	cell age	0.00	1	0.00	1	0.00
x_8	AE frequency	0.02	1	0.02	1	0.02
x_9	AE duration	0.02	1	0.02	1	0.02
x_{10}	un. anode change	0.26	1	0.26	1	0.26
x_{11}	rod height	0.61	1	0.61	1	0.61
x_{12}	cell power	0.04	1	0.04	1	0.04
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	1	0.01	1	0.01
x_{15}	cell resistance	0.03	1	0.03	1	0.03
x_{16}	emf	0.29	0	0.00	0	0.00
x_{17}	bath resistivity	0.02	1	0.02	1	0.02
x_{18}	cell voltage	0.01	1	0.01	1	0.01
x_{19}	AE energy	0.02	1	0.02	1	0.02
x_{20}	Si content	3.83	1	3.83	0	0.00
x_{21}	Fe content	3.83	1	3.83	0	0.00
x_{22}	Fe/V	3.83	1	3.83	0	0.00
x_{23}	Fe/Ga	3.83	1	3.83	0	0.00
x_{24}	high temp. count	1.15	1	1.15	1	1.15
x_{25}	low temp. count	1.15	1	1.15	1	1.15
x_{26}	computation time (hr)	0.51E-02	0.80	0.00	0.70	0.00
		TOTAL	-	57.26	-	51.26

TABLE F.11. Comparison of Original and Optimal Solutions for RBF Neural Network for Cell Failure Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.1279	64.16	0.1279	64.16
x_2	Fe content (weekly)	3.83	1	3.83	1	3.83
x_3	Si content (weekly)	3.83	1	3.83	1	3.83
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	0	0.00	0	0.00
x_6	bath temperature	1.15	1	1.15	1	1.15
x_7	cell age	0.00	1	0.00	1	0.00
x_8	AE frequency	0.02	1	0.02	1	0.02
x_9	AE duration	0.02	1	0.02	1	0.02
x_{10}	un. anode change	0.26	1	0.26	1	0.26
x_{11}	rod height	0.61	1	0.61	1	0.61
x_{12}	cell power	0.04	1	0.04	1	0.04
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	1	0.01	1	0.01
x_{15}	cell resistance	0.03	1	0.03	1	0.03
x_{16}	emf	0.29	0	0.00	0	0.00
x_{17}	bath resistivity	0.02	0	0.00	0	0.00
x_{18}	cell voltage	0.01	0	0.00	0	0.00
x_{19}	AE energy	0.02	1	0.02	1	0.02
x_{20}	Si content	3.83	0	0.00	0	0.00
x_{21}	Fe content	3.83	1	3.83	1	3.83
x_{22}	Fe/V	3.83	1	3.83	1	3.83

x_{23}	Fe/Ga	3.83	1	3.83	1	3.83
x_{24}	high temp. count	1.15	1	1.15	1	1.15
x_{25}	low temp. count	1.15	0	0.00	0	0.00
x_{26}	computation time (hr)	0.51E-02	0.50	0.00	0.50	0.00
		TOTAL	-	87.32	-	87.32

TABLE F.12. Comparison of Original and Optimal Solutions for RBFKOH Neural Network for Cell Failure Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.1074	53.88	0.1139	57.12
x_2	Fe content (weekly)	3.83	1	3.83	1	3.83
x_3	Si content (weekly)	3.83	0	0.00	0	0.00
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	0	0.00	0	0.00
x_6	bath temperature	1.15	1	1.15	1	1.15
x_7	cell age	0.00	1	0.00	1	0.00
x_8	AE frequency	0.02	1	0.02	1	0.02
x_9	AE duration	0.02	0	0.00	0	0.00
x_{10}	un. anode change	0.26	1	0.26	1	0.26
x_{11}	rod height	0.61	1	0.61	1	0.61
x_{12}	cell power	0.04	1	0.04	1	0.04
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	1	0.01	1	0.01
x_{15}	cell resistance	0.03	0	0.00	0	0.00
x_{16}	emf	0.29	0	0.00	0	0.00
x_{17}	bath resistivity	0.02	0	0.00	0	0.00
x_{18}	cell voltage	0.01	0	0.00	0	0.00
x_{19}	AE energy	0.02	1	0.02	1	0.02
x_{20}	Si content	3.83	0	0.00	0	0.00
x_{21}	Fe content	3.83	1	3.83	1	3.83
x_{22}	Fe/V	3.83	1	3.83	0	0.00
x_{23}	Fe/Ga	3.83	0	0.00	0	0.00
x_{24}	high temp. count	1.15	1	1.15	1	1.15
x_{25}	low temp. count	1.15	0	0.00	0	0.00
x_{26}	computation time (hr)	0.51E-02	1.26	0.01	1.24	0.01
		TOTAL	-	69.33	-	68.74

TABLE F.13. Comparison of Original and Optimal Solutions for GRNN for Cell Failure Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	501.68	0.1598	80.17	0.1650	82.79
x_2	Fe content (weekly)	3.83	1	3.83	0	0.00
x_3	Si content (weekly)	3.83	1	3.83	0	0.00
x_4	lining voltage drop	0.68	1	0.68	1	0.68
x_5	bath height	0.55	1	0.55	0	0.00
x_6	bath temperature	1.15	0	0.00	0	0.00
x_7	cell age	0.00	0	0.00	0	0.00
x_8	AE frequency	0.02	0	0.00	0	0.00
x_9	AE duration	0.02	0	0.00	0	0.00
x_{10}	un. anode change	0.26	0	0.00	0	0.00
x_{11}	rod height	0.61	0	0.00	0	0.00
x_{12}	cell power	0.04	0	0.00	0	0.00
x_{13}	high frequency noise	0.01	1	0.01	1	0.01
x_{14}	low frequency noise	0.01	1	0.01	1	0.01

x_{15}	cell resistance	0.03	0	0.00	0	0.00
x_{16}	emf	0.29	0	0.00	0	0.00
x_{17}	bath resistivity	0.02	0	0.00	0	0.00
x_{18}	cell voltage	0.01	1	0.01	1	0.01
x_{19}	AE energy	0.02	0	0.00	0	0.00
x_{20}	Si content	3.83	1	3.83	0	0.00
x_{21}	Fe content	3.83	1	3.83	0	0.00
x_{22}	Fe/V	3.83	1	3.83	0	0.00
x_{23}	Fe/Ga	3.83	0	0.00	0	0.00
x_{24}	high temp. count	1.15	1	1.15	0	0.00
x_{25}	low temp. count	1.15	1	1.15	0	0.00
x_{26}	computation time (hr)	0.51E-02	0.01	0.00	0.01	0.00
		TOTAL	-	102.88	-	83.50

TABLE F.14. Original Solution for Multi-Variable Regression Analysis (MVRA) Model for Cell Failure Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution	
			Units	Cost ($)
x_1	RMS error (test)	501.68	0.3264	163.75
x_2	Fe content (weekly)	3.83	1	3.83
x_3	Si content (weekly)	3.83	0	0.00
x_4	lining voltage drop	0.68	1	0.68
x_5	bath height	0.55	0	0.00
x_6	bath temperature	1.15	0	0.00
x_7	cell age	0.00	1	0.00
x_8	AE frequency	0.02	0	0.00
x_9	AE duration	0.02	0	0.00
x_{10}	un. anode change	0.26	1	0.26
x_{11}	rod height	0.61	0	0.00
x_{12}	cell power	0.04	0	0.00
x_{13}	high frequency noise	0.01	0	0.00
x_{14}	low frequency noise	0.01	1	0.01
x_{15}	cell resistance	0.03	0	0.00
x_{16}	emf	0.29	0	0.00
x_{17}	bath resistivity	0.02	0	0.00
x_{18}	cell voltage	0.01	0	0.00
x_{19}	AE energy	0.02	0	0.00
x_{20}	Si content	3.83	0	0.00
x_{21}	Fe content	3.83	1	3.83
x_{22}	Fe/V	3.83	1	3.83
x_{23}	Fe/Ga	3.83	0	0.00
x_{24}	high temp. count	1.15	1	1.15
x_{25}	low temp. count	1.15	0	0.00
x_{26}	computation time (hr)	0.51E-02	0.00	0.00
		TOTAL	-	177.34

TABLE F.15. Comparison of Original and Optimal Solutions for WH Neural Network for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Cost per Unit ($)	Original Solution		Optimal Solution	
			Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0626	44.38	0.0636	45.09
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	0	0.00	0	0.00
x_5	cell age	0.00	1	0.00	1	0.00
x_6	bath height	0.55	1	0.55	1	0.55

x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	1	0.01	1	0.01
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	1	1.17	0	0.00
x_{11}	Na_2CO_3 addition	1.17	1	1.17	1	1.17
x_{12}	computation time (hr)	0.70E-03	0.05	0.00	0.04	0.00
		TOTAL	-	48.50	-	48.04

TABLE F.16. Comparison of Original and Optimal Solutions for BP1 Neural Network for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0625	44.31	0.0625	44.31
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	0	0.00	0	0.00
x_5	cell age	0.00	0	0.00	0	0.00
x_6	bath height	0.55	0	0.00	0	0.00
x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	1	0.01	1	0.01
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	0	0.00	0	0.00
x_{11}	Na_2CO_3 addition	1.17	0	0.00	0	0.00
x_{12}	computation time (hr)	0.70E-03	0.18	0.00	0.18	0.00
		TOTAL	-	45.54	-	45.54

TABLE F.17. Comparison of Original and Optimal Solutions for BP2 Neural Network for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0573	40.63	0.0573	40.63
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	1	0.29	1	0.29
x_5	cell age	0.00	1	0.00	1	0.00
x_6	bath height	0.55	1	0.55	1	0.55
x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	1	0.01	1	0.01
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	1	1.17	1	1.17
x_{11}	Na_2CO_3 addition	1.17	1	1.17	1	1.17
x_{12}	computation time (hr)	0.70E-03	0.69	0.00	0.69	0.00
		TOTAL	-	45.04	-	45.04

TABLE F.18. Comparison of Original and Optimal Solutions for RBF Neural Network for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0665	47.15	0.0669	47.43
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	0	0.00	0	0.00
x_5	cell age	0.00	1	0.00	1	0.00

x_6	bath height	0.55	1	0.55	0	0.00
x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	0	0.00	0	0.00
x_9	cell power	0.04	0	0.00	0	0.00
x_{10}	AlF_3 addition	1.17	0	0.00	0	0.00
x_{11}	Na_2CO_3 addition	1.17	0	0.00	0	0.00
x_{12}	computation time (hr)	0.70E-03	0.25	0.00	0.25	0.00
		TOTAL	-	48.88	-	48.61

TABLE F.19. Comparison of Original and Optimal Solutions for RBFKOH Neural Network for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0640	45.38	0.0640	45.38
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	1	0.29	1	0.29
x_5	cell age	0.00	0	0.00	0	0.00
x_6	bath height	0.55	1	0.55	1	0.55
x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	0	0.00	0	0.00
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	0	0.00	0	0.00
x_{11}	Na_2CO_3 addition	1.17	0	0.00	0	0.00
x_{12}	computation time (hr)	0.70E-03	0.87	0.00	0.87	0.00
		TOTAL	-	47.44	-	47.44

TABLE F.20. Comparison of Original and Optimal Solutions for GRNN for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution		Optimal Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0702	49.77	0.0702	49.77
x_2	bath temperature (t-1)	1.15	1	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02	1	0.02
x_4	emf	0.29	1	0.29	1	0.29
x_5	cell age	0.00	1	0.00	1	0.00
x_6	bath height	0.55	1	0.55	1	0.55
x_7	high frequency noise	0.01	1	0.01	1	0.01
x_8	low frequency noise	0.01	1	0.01	1	0.01
x_9	cell power	0.04	1	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	0	0.00	0	0.00
x_{11}	Na_2CO_3 addition	1.17	0	0.00	0	0.00
x_{12}	computation time (hr)	0.70E-03	0.01	0.00	0.01	0.00
		TOTAL	-	51.84	-	51.84

TABLE F.21. Original Solution for Multi-Variable Regression Analysis (MVRA) Model for Electrolyte Temperature Prediction Application

Decision		Cost per	Original Solution	
Variable	Parameter Name	Unit ($)	Units	Cost ($)
x_1	RMS error (test)	709.00	0.0714	50.62
x_2	bath temperature (t-1)	1.15	1	1.15
x_3	bath resistivity	0.02	1	0.02
x_4	emf	0.29	1	0.29

x_5	cell age	0.00	1	0.00
x_6	bath height	0.55	1	0.55
x_7	high frequency noise	0.01	1	0.01
x_8	low frequency noise	0.01	1	0.01
x_9	cell power	0.04	1	0.04
x_{10}	AlF_3 addition	1.17	1	1.17
x_{11}	Na_2CO_3 addition	1.17	0	0.00
x_{12}	computation time (hr)	0.70E-03	0.00	0.00
		TOTAL	-	53.86

APPENDIX G

Confirmation of Optimal Solutions for Studied Industrial Applications

TABLE G.1. Comparison of Optimal and WH Neural Network Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.1172	0.1170
x_2	target bath temperature	1	1
x_3	bath temperature	0	0
x_4	bath height	0	0
x_5	bath resistivity	1	1
x_6	emf	0	0
x_7	AlF_3 addition (t-1)	0	0
x_8	Na_2CO_3 addition (t-1)	0	0
x_9	cell power	0	0
x_{10}	cell age	0	0
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.06	0.06

TABLE G.2. Comparison of Optimal and BP1 Neural Network Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0804	0.0807
x_2	target bath temperature	1	1
x_3	bath temperature	1	1
x_4	bath height	0	0
x_5	bath resistivity	1	1
x_6	emf	0	0
x_7	AlF_3 addition (t-1)	0	0
x_8	Na_2CO_3 addition (t-1)	1	1
x_9	cell power	1	1
x_{10}	cell age	1	1
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.24	0.24

TABLE G.3. Comparison of Optimal and BP2 Neural Network Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0873	0.0872
x_2	target bath temperature	1	1
x_3	bath temperature	1	1
x_4	bath height	1	1
x_5	bath resistivity	1	1
x_6	emf	1	1
x_7	AlF_3 addition (t-1)	1	1
x_8	Na_2CO_3 addition (t-1)	0	0
x_9	cell power	1	1
x_{10}	cell age	1	1
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.77	0.77

TABLE G.4. Comparison of Optimal and RBF Neural Network Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0755	0.0757
x_2	target bath temperature	1	1
x_3	bath temperature	1	1
x_4	bath height	0	0
x_5	bath resistivity	1	1
x_6	emf	0	0
x_7	AlF_3 addition (t-1)	1	1
x_8	Na_2CO_3 addition (t-1)	0	0
x_9	cell power	1	1
x_{10}	cell age	1	1
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.31	0.31

TABLE G.5. Comparison of Optimal and RBFKOH Neural Network Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0723	0.0722
x_2	target bath temperature	1	1
x_3	bath temperature	1	1
x_4	bath height	0	0
x_5	bath resistivity	1	1
x_6	emf	0	0
x_7	AlF_3 addition (t-1)	1	1
x_8	Na_2CO_3 addition (t-1)	0	0
x_9	cell power	1	1
x_{10}	cell age	1	1
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.73	0.73

TABLE G.6. Comparison of Optimal and GRNN Solution for Electrolyte Additive Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0742	0.0743
x_2	target bath temperature	1	1
x_3	bath temperature	0	0
x_4	bath height	0	0
x_5	bath resistivity	1	1
x_6	emf	0	0
x_7	AlF_3 addition (t-1)	0	0
x_8	Na_2CO_3 addition (t-1)	0	0
x_9	cell power	0	0
x_{10}	cell age	0	0
x_{11}	F content	0	0
x_{12}	Na content	0	0
x_{13}	temperature reference	1	1
x_{14}	computation time (hr)	0.01	0.01

TABLE G.7. Comparison of Optimal and WH Neural Network Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0818	0.0815
x_2	Fe content (weekly)	0	0
x_3	Si content (weekly)	0	0
x_4	lining voltage drop	1	1
x_5	bath height	0	0
x_6	bath temperature	0	0
x_7	cell age	1	1
x_8	AE frequency	1	1
x_9	AE duration	1	1
x_{10}	unscheduled anode change	1	1
x_{11}	rod height	0	0
x_{12}	cell power	1	1
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	0	0
x_{15}	cell resistance	1	1
x_{16}	emf	0	0
x_{17}	bath resistivity	1	1
x_{18}	cell voltage	1	1
x_{19}	AE energy	1	1
x_{20}	Si content	0	0
x_{21}	Fe content	0	0
x_{22}	Fe/V	0	0
x_{23}	Fe/Ga	0	0
x_{24}	high temperature count	1	1
x_{25}	low temperature count	0	0
x_{26}	computation time (hr)	0.02	0.02

TABLE G.8. Comparison of Optimal and BP1 Neural Network Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0451	0.0451
x_2	Fe content (weekly)	1	1
x_3	Si content (weekly)	1	1

x_4	lining voltage drop	1	1
x_5	bath height	1	1
x_6	bath temperature	1	1
x_7	cell age	1	1
x_8	AE frequency	1	1
x_9	AE duration	1	1
x_{10}	unscheduled anode change	1	1
x_{11}	rod height	1	1
x_{12}	cell power	1	1
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	1	1
x_{15}	cell resistance	1	1
x_{16}	emf	1	1
x_{17}	bath resistivity	1	1
x_{18}	cell voltage	1	1
x_{19}	AE energy	1	1
x_{20}	Si content	1	1
x_{21}	Fe content	1	1
x_{22}	Fe/V	1	1
x_{23}	Fe/Ga	1	1
x_{24}	high temperature count	1	1
x_{25}	low temperature count	0	0
x_{26}	computation time (hr)	1.23	1.23

TABLE G.9. Comparison of Optimal and BP2 Neural Network Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0789	0.0787
x_2	Fe content (weekly)	1	1
x_3	Si content (weekly)	1	1
x_4	lining voltage drop	1	1
x_5	bath height	0	0
x_6	bath temperature	0	0
x_7	cell age	1	1
x_8	AE frequency	1	1
x_9	AE duration	1	1
x_{10}	unscheduled anode change	1	1
x_{11}	rod height	1	1
x_{12}	cell power	1	1
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	1	1
x_{15}	cell resistance	1	1
x_{16}	emf	0	0
x_{17}	bath resistivity	1	1
x_{18}	cell voltage	1	1
x_{19}	AE energy	1	1
x_{20}	Si content	0	0
x_{21}	Fe content	0	0
x_{22}	Fe/V	0	0
x_{23}	Fe/Ga	0	0
x_{24}	high temperature count	1	1
x_{25}	low temperature count	1	1
x_{26}	computation time (hr)	0.70	0.70

TABLE G.10. Comparison of Optimal and RBF Neural Network Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.1279	0.1279
x_2	Fe content (weekly)	1	1
x_3	Si content (weekly)	1	1
x_4	lining voltage drop	1	1
x_5	bath height	0	0
x_6	bath temperature	1	1
x_7	cell age	1	1
x_8	AE frequency	1	1
x_9	AE duration	1	1
x_{10}	unscheduled anode change	1	1
x_{11}	rod height	1	1
x_{12}	cell power	1	1
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	1	1
x_{15}	cell resistance	1	1
x_{16}	emf	0	0
x_{17}	bath resistivity	0	0
x_{18}	cell voltage	0	0
x_{19}	AE energy	1	1
x_{20}	Si content	0	0
x_{21}	Fe content	1	1
x_{22}	Fe/V	1	1
x_{23}	Fe/Ga	1	1
x_{24}	high temperature count	1	1
x_{25}	low temperature count	0	0
x_{26}	computation time (hr)	0.50	0.50

TABLE G.11. Comparison of Optimal and RBFKOH Neural Network Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.1139	0.1142
x_2	Fe content (weekly)	1	1
x_3	Si content (weekly)	0	0
x_4	lining voltage drop	1	1
x_5	bath height	0	0
x_6	bath temperature	1	1
x_7	cell age	1	1
x_8	AE frequency	1	1
x_9	AE duration	0	0
x_{10}	unscheduled anode change	1	1
x_{11}	rod height	1	1
x_{12}	cell power	1	1
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	1	1
x_{15}	cell resistance	0	0
x_{16}	emf	0	0
x_{17}	bath resistivity	0	0
x_{18}	cell voltage	0	0
x_{19}	AE energy	1	1
x_{20}	Si content	0	0
x_{21}	Fe content	1	1
x_{22}	Fe/V	0	0
x_{23}	Fe/Ga	0	0
x_{24}	high temperature count	1	1

x_{25}	low temperature count	0	0
x_{26}	computation time (hr)	1.24	1.24

TABLE G.12. Comparison of Optimal and GRNN Solution for Cell Failure Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.1650	0.1648
x_2	Fe content (weekly)	0	0
x_3	Si content (weekly)	0	0
x_4	lining voltage drop	1	1
x_5	bath height	0	0
x_6	bath temperature	0	0
x_7	cell age	0	0
x_8	AE frequency	0	0
x_9	AE duration	0	0
x_{10}	unscheduled anode change	0	0
x_{11}	rod height	0	0
x_{12}	cell power	0	0
x_{13}	high frequency noise	1	1
x_{14}	low frequency noise	1	1
x_{15}	cell resistance	0	0
x_{16}	emf	0	0
x_{17}	bath resistivity	0	0
x_{18}	cell voltage	1	1
x_{19}	AE energy	0	0
x_{20}	Si content	0	0
x_{21}	Fe content	0	0
x_{22}	Fe/V	0	0
x_{23}	Fe/Ga	0	0
x_{24}	high temperature count	0	0
x_{25}	low temperature count	0	0
x_{26}	computation time (hr)	0.01	0.01

TABLE G.13. Comparison of Optimal and WH Neural Network Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0636	0.0639
x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	0	0
x_5	cell age	1	1
x_6	bath height	1	1
x_7	high frequency noise	1	1
x_8	low frequency noise	1	1
x_9	cell power	1	1
x_{10}	AlF_3 addition	0	0
x_{11}	Na_2CO_3 addition	1	1
x_{12}	computation time (hr)	0.04	0.04

TABLE G.14. Comparison of Optimal and BP1 Neural Network Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0625	0.0625

x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	0	0
x_5	cell age	0	0
x_6	bath height	0	0
x_7	high frequency noise	1	1
x_8	low frequency noise	1	1
x_9	cell power	1	1
x_{10}	AlF_3 addition	0	0
x_{11}	Na_2CO_3 addition	0	0
x_{12}	computation time (hr)	0.18	0.18

TABLE G.15. Comparison of Optimal and BP2 Neural Network Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0573	0.0573
x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	1	1
x_5	cell age	1	1
x_6	bath height	1	1
x_7	high frequency noise	1	1
x_8	low frequency noise	1	1
x_9	cell power	1	1
x_{10}	AlF_3 addition	1	1
x_{11}	Na_2CO_3 addition	1	1
x_{12}	computation time (hr)	0.69	0.69

TABLE G.16. Comparison of Optimal and RBF Neural Network Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0669	0.0671
x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	0	0
x_5	cell age	1	1
x_6	bath height	0	0
x_7	high frequency noise	1	1
x_8	low frequency noise	0	0
x_9	cell power	0	0
x_{10}	AlF_3 addition	0	0
x_{11}	Na_2CO_3 addition	0	0
x_{12}	computation time (hr)	0.25	0.25

TABLE G.17. Comparison of Optimal and RBFKOH Neural Network Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0640	0.0640
x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	1	1
x_5	cell age	0	0

x_6	bath height	1	1
x_7	high frequency noise	1	1
x_8	low frequency noise	0	0
x_9	cell power	1	1
x_{10}	AlF_3 addition	0	0
x_{11}	Na_2CO_3 addition	0	0
x_{12}	computation time (hr)	0.87	0.87

TABLE G.18. Comparison of Optimal and GRNN Solution for Electrolyte Temperature Prediction Application

Decision Variable	Parameter Name	Optimal Solution (units)	Neural Network (units)
x_1	RMS error (test)	0.0702	0.0702
x_2	bath temperature (t-1)	1	1
x_3	bath resistivity	1	1
x_4	emf	1	1
x_5	cell age	1	1
x_6	bath height	1	1
x_7	high frequency noise	1	1
x_8	low frequency noise	1	1
x_9	cell power	1	1
x_{10}	AlF_3 addition	0	0
x_{11}	Na_2CO_3 addition	0	0
x_{12}	computation time (hr)	0.01	0.01

APPENDIX H

Relevant Publications

A full document for each the publications listed in the following is available on the accompanying software in the directory *Publications*. The papers are stored appropriately in the sub-directory corresponding to the conference or journal title.

[1]. Karri, V. and Frost, F., *An Intelligent System for Detection of Failed Aluminium Wheels*, Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), February 1997, Gold Coast, QLD, Australia, pp. 107-111.

[2]. Frost, F. and Karri, V., *Behaviour of 601-Alloy Using Two Different Solution Treatment Methods: Part I - Mechanical Properties*, Proc. Materials 98 Conference (MATS), July 1998, Wollongong, NSW, Australia, pp. 597-602.

[3]. Karri, V. and Frost, F., *Behaviour of 601-Alloy Using Two Different Solution Treatment Methods: Part II - Machining Characteristics*, Proc. Materials 98 Conference (MATS), July 1998, Wollongong, NSW, Australia, pp. 603-608.

[4]. Frost, F. and Karri, V., *Effect of Heat Treatment on Mechanical Properties and Surface Roughness of Aluminium Alloy 601*, Journal of Testing and Evaluation, vol. 27, no. 4, July 1999, pp. 273-281.

[5]. Frost, F. and Karri, V., *Determining the Influence of Input Parameters on BP Neural Network Output Error Using Sensitivity Analysis*, Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), September 1999, New Delhi, India, pp. 45-49.

[6]. Karri, V. and Frost, F., *Optimum Backpropagation Network Conditions with Respect to Computation Time and Output Accuracy*, Proc. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA), September 1999, New Delhi, India, pp. 50-54.

[7]. Karri, V. and Frost, F., *Effect of Altering the Gaussian Function Receptive Field Width in RBF Neural Networks on Aluminium Fluoride Prediction in Industrial Reduction Cells*, Proc. International Conference on Neural Information Processing (ICONIP), November 1999, Perth, WA, Australia, pp. 101-106.

[8]. Frost, F. and Karri, V., *Performance Comparison of BP and GRNN Models of the Neural Network Paradigm Using a Practical Industrial Application*, Proc. International Conference on Neural Information Processing (ICONIP), November 1999, Perth, WA, Australia, pp. 1069-1074.

[9]. Karri, V. and Frost, F., *Combined Kohonen and RBF Networks to Predict Electrolyte Additives in Hall-Heroult Cell*, Proc. International Conference on Advances in Intelligent Systems: Theory and Applications (AISTA), February 2000, Canberra, ACT, Australia, pp. 13-18.

[10]. Frost, F. and Karri, V., *Intelligent Control of Aluminium Reduction Cells Using Backpropagation Neural Networks*, Proc. International Conference on Advances in Intelligent Systems: Theory and Applications (AISTA), February 2000, Canberra, ACT, Australia, pp. 34-39.

[11]. Frost, F. and Karri, V., *Identifying Significant Parameters for Hall-Heroult Process Modelling using General Regression Neural Network*, Proc. Thirteenth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, (IEA/AIE), New Orleans, Louisiana, USA, June 2000, pp. 73-78.

[12]. Frost, F. and Karri, V., *Productivity Improvements Through Prediction of Electrolyte Temperature in Aluminium Reduction Cell Using BP Neural Network*, Proc. Sixth Pacific Rim International Conference on Artificial Intelligence (PRICAI), August 2000, Melbourne, VIC, Australia, pp. 480-489.

[13]. Karri, V. and Frost, F., *Need for Optimisation Techniques to Select Neural Network Algorithms for Process Modelling of Reduction Cell*, Proc. Sixth Pacific Rim International Conference on Artificial Intelligence (PRICAI), August 2000, Melbourne, VIC, Australia, pp. 490-499.

[14]. Frost, F. and Karri, V., *Incorporating Operations Research Techniques to Maximise Economic Benefit of Neural Network Modelling*, Proc. International Conference on Artificial Intelligence in Science and Technology (AISAT), December 2000, Hobart, TAS, Australia.

[15]. V. Karri and Frost, F., *Application of General Regression Neural Network for Estimation and Control in a Dynamic Industry Environment,* Proc. International Conference on Artificial Intelligence in Science and Technology (AISAT), December 2000, Hobart, TAS, Australia.